entropy

MDPI

Article

Dynamic Virtual Network Reconfiguration Method for Hybrid Multiple Failures Based on Weighted Relative Entropy

Yuze Su *, Xiangru Meng, Qiaoyan Kang and Xiaoyang Han

College of Information and Navigation, Air Force Engineering University, Xi'an 710077, China; xiangr_meng@163.com (X.M.); qiaoy_kang@163.com (Q.K.); dodogg1898@163.com (X.H.)

* Correspondence: wglxl@nudt.edu.cn; Tel.: +86-155-9480-5937

Received: 9 August 2018; Accepted: 13 September 2018; Published: 15 September 2018

Abstract: Network virtualization can offer more flexibility and better manageability for next generation Internet. With the increasing deployments of virtual networks in military and commercial networks, a major challenge is to ensure virtual network survivability against hybrid multiple failures. In this paper, we study the problem of recovering virtual networks affected by hybrid multiple failures in substrate networks and provide an integer linear programming formulation to solve it. We propose a heuristic algorithm to tackle the complexity of the integer linear programming formulation, which includes a faulty virtual network reconfiguration ranking method based on weighted relative entropy, a hybrid multiple failures ranking algorithm, and a virtual node migration method based on weighted relative entropy. In the faulty virtual network reconfiguration ranking method based on weighted relative entropy and virtual node migration method based on weighted relative entropy, multiple ranking indicators are combined in a suitable way based on weighted relative entropy. In the hybrid multiple failures ranking algorithm, the virtual node and its connective virtual links are re-embedded, firstly. Evaluation results show that our heuristic method not only has the best acceptance ratio and normal operation ratio, but also achieves the highest long-term average revenue to cost ratio compared with other virtual network reconfiguration methods.

Keywords: virtual network; weighted relative entropy; reconfiguration; hybrid multiple failures; survivability

1. Introduction

Network virtualization (NV) allows multiple heterogeneous virtual networks (VNs) to be embedded onto the shared substrate network (SN), providing users with a variety of network services which has become one of the most promising technologies for future Internet [1–3]. It can decouple the network infrastructure and network services, and allow multiple heterogeneous VNs to share the SN resources through abstraction, distribution and isolation mechanism. In recent years, NV has been widely used in various fields, such as fog-supported software defined network (SDN) [4] and virtualized networked data centers [5]. One of the most challenging problems raised in NV is the virtual network embedding (VNE) problem, which has been known as the non-deterministic polynomial hard (NP-hard) problem [6].

In recent years, the SN failure events occur frequently. SN failures degrade the service performance and reliability of VNs [7]. SN failures include node failure and link failure. In network virtualization, multiple VNs are embedded onto the same SN and share the common substrate resources. Hence, substrate node failure causes virtual node and its adjacent virtual link failure. Substrate link failure causes virtual link failure. Also, there are single node/link failures and multiple node/link failures

in SN which will cause complex and multiple virtual network (VN) failures [8]. Therefore, adopting suitable survivability strategies to against the increasingly complex network failures has become one of the main challenges in VN survivability.

A number of mechanisms have been proposed to improve the VN survivability [9]. These mechanisms can be broadly classified into two categories: backup mechanism [10] and reconfiguration mechanism [11]. The backup mechanism can also be classified into dedicated backup [12] and shared backup [13]. In shared backup mechanism, the backup resources can be shared with different VNs. In the dedicated case, the backup resources only can be used for dedicated VN. However, preallocating a backup resource is extremely expensive.

Instead, the reconfiguration mechanism can re-embed the faulty part of a VN without huge backup resource consumption. In reconfiguration mechanisms, backup resources are not preallocated, and the faulty parts of VNs are re-embedded after SN failures [14]. For instance, a topology awareness VN reconfiguration algorithm is proposed in [15]. The virtual node to be migrated is mapped onto the nearest mapped substrate node which is the neighbor mapping node of the migration virtual node. This algorithm takes topology indicator into consideration and improves the resource utilization. However, resource indicator is not considered in its node re-embedding strategy. To survive link failures, a novel recovery approach to restore VN without reserving backup resources prior to embedding is proposed in [16]. It not only considers the end-delay and delay variation requirements, but also takes actions to find a constrained shortest path between the nodes utilized by the failed VNs. The resource utilization is improved. However, this algorithm cannot deal with the problem of recovering VNs from node failures. As can be seen from above two algorithms, they cannot deal with the hybrid failures which include node failures and link failures.

To survive hybrid failures in SN, a heuristic survivable VNE based on node migration and link remapping is proposed in [17]. In VNE, the artificial bee colony algorithm is proposed to achieve an optimal solution. If the substrate node fails, the failed node is migrated to a normal node based on the greed rules, and the affected links are remapped based on the shortest path algorithm. Then, the shortest path is used to provide the best bandwidth principle to remap the affected links that connect with the failed node. This algorithm can deal with hybrid failures. However, the greed rules only take resource indicator into consideration and their re-embedding performance is limited. Also, they do not consider the hybrid failures in multiple faulty VNs.

To solve the problem of recovering a batch of VNs affected by a substrate node failure, a recovery approach for maximizing recovery and minimizing the cost of recovery is designed in [18]. An integer linear programming (ILP) formulation of this recovery scheme is provided, and a fast and scalable heuristic algorithm is also proposed to tackle the computational complexity of the ILP solution. It can recover a batch of VNs affected by a substrate node failure efficiently. However, it sorts the faulty VNs and virtual nodes based on the adjacent bandwidth resource, and it ignores the topology indicators. A generalized recovery approach that can achieve customized objectives is designed, and the corresponding ILP formulation is provided in [19]. Then, a fast and scalable heuristic algorithm to tackle the computational complexity of the ILP solution is proposed, and it is demonstrated that this heuristic algorithm has good recovery performance. However, it focuses on the problem of recovering from a node failure in VNE and ranks faulty VNs based on service level agreements. It cannot deal with the problem of recovering a batch of VNs affected by hybrid failures which include node and link failures. Also, in different recovery objects, recovering the faulty VNs based on service level agreements is not suitable.

As can be seen from the current research, the VN reconfiguration mechanisms focus on taking effective measures for network recovery after failures without preallocating resources. Although numerous efforts have been devoted, there are four obvious defects in existing VN reconfiguration algorithms. At first, VN reconfiguration for different types of failures in one VN and VN reconfiguration for single type of failures in multiple VNs are both studied. However, the VN reconfiguration for multiple types of failures in multiple VNs is not studied. Secondly,

in reconfiguration of multiple faulty VNs, a single indicator is selected in most faulty VN ranking strategies. In different recovery objects, these strategies are not always suitable. Thirdly, a single resource indicator or topology indicator is taken into consideration in VN re-embedding, and the performance is limited. At last, the failures are not classified into certain types, and the recovery order of different types of failures is not researched.

In this paper, an ILP formulation of VN reconfiguration is provided, firstly. Then, the heuristic algorithm, called dynamic virtual network reconfiguration method for hybrid multiple failures based on weighted relative entropy (DVNRM-HMF-WRE), is proposed to solve the ILP formulation. In DVNRM-HMF-WRE, the faulty VN reconfiguration ranking method based on weighted relative entropy (FVNRRM-WRE), hybrid multiple failures ranking algorithm (HMFRA), and virtual node migration method based on weighted relative entropy (VNMM-WRE), are proposed. At last, five comparative experiments are set to demonstrate the performance of DVNRM-HMF-WRE.

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

(1) We provide an ILP formulation of VN reconfiguration and propose the DVNRM-HMF-WRE to solve it. VNE is already intractable, and the combinatorial number of sequences of VNs in batch re-embedding further increases the complexity. Therefore, we only re-embed the faulty nodes and links without disrupting their unaffected parts in VN.
(2) We introduce a weighted relative entropy (WRE) method into FVNRRM-WRE and VNMM-WRE. In FVNRRM-WRE, we rank the faulty VNs with the help of multiple indicators, and combine them based on WRE. In VNMM-WRE, we use the WRE method to change the coefficients of node resource, and topology indicator to select the suitable candidate substrate node to reduce the resource consumption.
(3) We propose the HMFRA to handle the hybrid multiple failures. In HMFRA, faulty virtual node and its connective virtual links are handled first, which are the hardest the recover. HMFRA can make full use of the limited resources and improve the recovery performance.
(4) Five different simulation scenarios are conducted to validate the performance of DVNRM-HMF-WRE. The performance of DVNRM-HMF-WRE is compared with other algorithms in the first simulation experiment. Next, we simulate the impact of FVNRRM-WRE, HMFRA, and VNMM-WRE on the performance of DVNRM-HMF-WRE. At last, two factors are set to evaluate their influences on DVNRM-HMF-WRE.

The rest of this paper is organized as follows. Section 2 presents the models and evaluation indicators. In Section 3, we propose the ILP of VN reconfiguration. The DVNRM-HMF-WRE is given, and its details are shown in Section 4. In Section 5, we evaluate the proposed algorithm through extensive simulations and experiments. We conclude this paper with a summary and areas for future exploration in Section 6.

2. Models and Evaluation Indicators

In this section, we describe the models and evaluation indicators. Here, Table 1 summarizes the main notations used in this paper.

Table 1. Notations.

Notation	Description	Notation	Description
$\boldsymbol{G}_S$	SN	$\boldsymbol{G}_V$	VN
N_S	Substrate node set	N_V	Virtual node set
E_S	Substrate link set	E_V	Virtual link set
$\mid N_S \mid$	Number of substrate nodes	$\mid N_V \mid$	Number of virtual nodes
$\mid E_S \mid$	Number of substrate links	$\mid E_V \mid$	Number of virtual links
n_s	Substrate node	n_v	Virtual node
e_s	Substrate link	e_v	Virtual link
$cpu(n_s)$	CPU resource of n_s	$cpu(n_v)$	Required CPU resource of n_v
$bw(e_s)$	Bandwidth resource of e_s	$bw(e_v)$	Required bandwidth resource of e_v
$loc(n_s)$	Location attribute of n_s	N_{Vf}	Faulty virtual node set
N_{Sf}	Faulty substrate node set	E_{Vf}	Faulty virtual link set
E_{Sf}	Faulty substrate link set	T_V	Lifetime of VN
$\boldsymbol{G}_{Vf}$	Faulty VN set	$F_{i,j}$	Failure events
$F_{1,j}$	Substrate node failure	$F_{2,j}$	Substrate link failure
$t_e(F_{i,j})$	Ending time of $F_{i,j}$	$t_s(F_{i,j})$	Starting time of $F_{i,j}$
$VN_{map}(t)$	VNs which are embedded successfully at time t	$\mid VN_{map}(t) \mid$	Number of VNs which are embedded successfully at time t
$R(\boldsymbol{G}_V, t)$	Revenue of $\boldsymbol{G}_V$ at time t	$VN_{fr}(t)$	Set of faulty VNs which are re-embedded unsuccessfully
$C(\boldsymbol{G}_V, t)$	Cost of $\boldsymbol{G}_V$ at time t	$PCost(\boldsymbol{G}_V, t)$	Penalty cost of $\boldsymbol{G}_V$ at time t
$hops(e_v)$	Hop counts in substrate link corresponding to virtual link e_v	$\mid VN_{fr}(t) \mid$	Number of faulty VNs which are re-embedded unsuccessfully
$\mid VN(t) \mid$	Total number of VN requests at time t	$\eta(t)$	Normal operation ratio at time t
$x(n_v,n_s)\in\{0,1\}$	$x(n_v, n_s) = 1$ if virtual node n_v is embedded onto substrate node n_s	$x(e_{up}, e_{ij})\in\{0,1\}$	$x(e_{up}, e_{ij}) = 1$ if virtual link e_{up} is embedded onto substrate link e_{ij}
$s(n_v)\in\{0,1\}$	$s(n_v) = 1$ if n_v is faulty	$s(n_s)\in\{0,1\}$	$s(n_s) = 1$ if n_s is faulty
$s(e_v)\in\{0,1\}$	$s(e_v) = 1$ if e_v is faulty	$s(e_s)\in\{0,1\}$	$s(e_s) = 1$ if e_s is faulty
$s(\boldsymbol{G}_V)\in\{0,1\}$	$s(\boldsymbol{G}_V) = 1$ if $\boldsymbol{G}_V$ is faulty	$s(\boldsymbol{G}_S)\in\{0,1\}$	$s(\boldsymbol{G}_S) = 1$ if $\boldsymbol{G}_S$ is faulty
$dis(loc(n_s), loc(n_v))$	Distance between n_s and n_v	A_i	State of system A
B_i	State of system B	C	Entropy value
X_{ij}	The j-th indicator coefficient of the ith node	$\boldsymbol{X}$	Decision matrix
ω_j	Weighting coefficient of the j-th indicator	A^+	Positive ideal solutions
A^-	Negative ideal solutions	Z_i	Similarity between each solution and the ideal one
$Z(n_v)$	Importance degree of virtual node n_v	$Z(n_s)$	Importance degree of substrate node n_s
$Can(n_{vi})$	Candidate substrate node set	$\mid N_{FN} \mid$	Total number of faulty VNs

2.1. Network Model

2.1.1. 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 substrate nodes, the available CPU resource of substrate node n_s is denoted by $cpu(n_s)$, and the location attribute of substrate node n_s is denoted by $loc(n_s)$. Similarly, $bw(e_s)$ is taken to denote the available bandwidth resource of substrate link e_s. Also, N_{Sf} represents faulty substrate node set and E_{Sf} represents the faulty substrate link set.

2.1.2. Virtual Network

Similar to SN, the VN can be modeled as a weighted undirected graph $\boldsymbol{G}_V = (N_V, E_V)$. N_V represents virtual node set and E_V represents the virtual link set. In virtual nodes, $cpu(n_v)$ denotes the required CPU resource of virtual node n_v. In virtual links, $bw(e_v)$ is taken as the basic attribute to denote the required bandwidth resource of virtual link e_v. T_V is used to denote the VN lifetime. $\boldsymbol{G}_{Vf}$ represents the faulty VN set. Also, N_{Vf} represents faulty virtual node set and E_{Vf} represents the faulty virtual link set.

2.2. Failure Model

2.2.1. Multiple Failures

The SN failures are denoted by a series of failure events $F_{i,j}$, I = 1,2; j = 1, 2,...,n. $F_{1,j}$ denotes the substrate node failure and $F_{1,j} \in N_{Sf}$. $F_{2,j}$ denotes the substrate link failure and $F_{2,j} \in E_{Sf}$, j is the failure number. $t_s(F_{i,j})$ and $t_e(F_{i,j})$ denote the starting and ending time of $F_{i,j}$, and $t_s(F_{i,j}) < t_e(F_{i,j})$. When $\exists j1 < j2$, if $t_s(F_{i,j1}) < t_s(F_{i,j2}) < t_e(F_{i,j1})$, multiple failures happen in SN. Multiple failures contain multiple node failures and multiple link failures.

2.2.2. Failure Types

VN failures caused by SN failures can be divided into virtual node failures and virtual link failures. Also, virtual link failures can be divided into three categories. If virtual link takes faulty substrate node as the source or destination node, it is called VN link failure I. The virtual link which passes through the faulty substrate node but does not take it as the source or destination node, is called VN link failure II. The independent virtual link failure caused by independent substrate link failure is called VN link failure III.

2.2.3. Hybrid Multiple Failures

If multiple failures occur at the same time and belong to multiple types, they are called hybrid multiple failures.

The process of VNE and VN reconfiguration are shown in Figure 1. Figure 1a shows two VNs, and Figure 1b illustrates an example for VNE problem. In VN1, the node embedding results are {a→A, b→B, c→C}, and link embedding results are {(a,b)→(A,B), (a,c)→(A,C)}, in which both the CPU and bandwidth requirements are satisfied. In VN2, the node embedding results are {d→H, e→G, f→I} and link embedding results are {(d,e)→(H,G), (d,f)→(H,I), (e,f) →(G,D,B,A,C,E,I)}.

Figure 1c,d illustrate the examples for VN reconfiguration. In Figure 1c, the substrate node A is faulty, which causes the connected substrate link (A,B) and (A,C) failure. The faulty virtual link (a,b) and (a,c) belongs to VN link failure I. Also, virtual link (e,f) in VN2 is faulty, which belongs to VN link failure II. At this time, virtual node a is re-embedded onto substrate node D. Virtual link (a,b) and (a,c) are re-embedded onto substrate link (D,B) and (D,F,E,C), respectively. After that, VN1 returns to normal. Virtual link (e,f) is re-embedded onto substrate link (G,D,F,E,I), then VN2 returns to normal.

In Figure 1d, substrate link (B,D) is faulty, which causes the virtual link (e,f) failure and this belongs to VN link failure III. After re-embedding virtual link (e,f) onto substrate link (G,D,F,E,I), VN2 returns to normal.

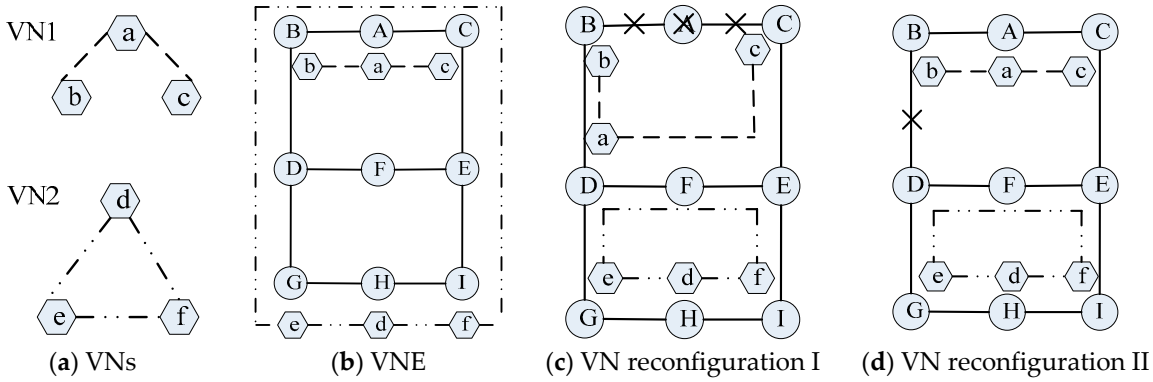

Figure 1. Examples of VN embedding and reconfiguration. (**a**) The VNs; (**b**) The process of VNE; (**c**) The process of VN reconfiguration for VN node failure, VN link failure I and II; (**d**) The process of VN reconfiguration for VN link failure III.

2.3. Evaluation Indicator

2.3.1. Acceptance Ratio

The acceptance ratio (AR) is denoted by the number of VN requests which are embedded successfully, divided by the total number of VN requests. It is shown in Formula (1).

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

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

2.3.2. Revenue to Cost Ratio

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

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

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

where α and β are weighting coefficients to balance CPU and bandwidth resources. In this paper, we set $\alpha = \beta = 1$. Also, $hops(e_{\text{v}})$ is the hop counts in substrate link corresponding to virtual link e_{v}.

At time t_i, if $\boldsymbol{G}_{\text{V}}$ is faulty and can be re-embedded successfully, the revenue and cost will not change. Otherwise, the revenue and cost of $\boldsymbol{G}_{\text{V}}$ are set to zero, and it also causes the penalty cost *PCost* which can be defined as

$$PCost(\boldsymbol{G}_{\text{V}}, t) = \mu \cdot (t_j - t_i) \cdot R(\boldsymbol{G}_{\text{V}}, t), \tag{4}$$

where t_j is terminal time of $\boldsymbol{G}_{\text{V}}$ and μ is the penalty coefficient. In this paper, $\mu = 2$.

Long-term average revenue to cost ratio (LAR/CR) can be defined as

$$\xi = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} R(\boldsymbol{G}_{\text{V}}, t) - \sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{fr}}(t)} PCost(\boldsymbol{G}_{\text{V}}, t)}{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} C(\boldsymbol{G}_{\text{V}}, t)}, \tag{5}$$

where $VN_{\text{fr}}(t)$ is the set of faulty VNs which are re-embedded unsuccessfully. $VN_{\text{map}}(t)$ is the VNs which are embedded successfully at time t.

2.3.3. Normal Operation Ratio

At time t, $|VN_{\text{fr}}(t)|$ denotes the number of faulty VNs which are unsuccessfully re-embedded. $|VN(t)|$ is the total number of embedded VN requests. The normal operation ratio (NOR) can be defined as

$$\eta(t) = \frac{|VN(t)| - |VN_{\text{fr}}(t)|}{|VN(t)|}. \tag{6}$$

3. ILP of VN Reconfiguration

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

3.1. Objective Function

$$\max\{\lim_{T\to\infty}\frac{\sum_{t=0}^{T}\sum_{\boldsymbol{G}_{\mathrm{V}}\subset VN_{\mathrm{map}}(t)}R(\boldsymbol{G}_{\mathrm{V}},t)-\sum_{t=0}^{T}\sum_{\boldsymbol{G}_{\mathrm{V}}\subset VN_{\mathrm{fr}}(t)}PCost(\boldsymbol{G}_{\mathrm{V}},t)}{\sum_{t=0}^{T}\sum_{\boldsymbol{G}_{\mathrm{V}}\subset VN_{\mathrm{map}}(t)}C(\boldsymbol{G}_{\mathrm{V}},t)}\}. \tag{7}$$

In this paper, our object is to get the maximum long-term average revenue to cost ratio. The corresponding variables in Formula (7) can be seen from Formulas (2)–(5) or Table 1.

3.2. Constraints

$$x(n_{\mathrm{v}},n_{\mathrm{s}})=\begin{cases}1\ iff\ n_{\mathrm{v}}\text{ is re-embedded onto }n_{\mathrm{s}}\\0\text{ otherwise}\end{cases},\quad \forall n_{\mathrm{v}}\in N_{\mathrm{V}},\ \forall n_{\mathrm{s}}\in N_{\mathrm{S}}, \tag{8}$$

$$x(e_{up},e_{ij})=\begin{cases}1\ iff\ e_{up}\text{ is re-embedded onto }e_{ij}\\0\text{ otherwise}\end{cases},\quad \forall e_{up}\in E_{\mathrm{V}},\ \forall e_{ij}\in E_{\mathrm{S}}, \tag{9}$$

$$s(n_{\mathrm{v}})=\begin{cases}1\ iff\ n_{\mathrm{v}}\text{ is faulty}\\0\text{ otherwise}\end{cases},\ s(n_{\mathrm{s}})=\begin{cases}1\ iff\ n_{\mathrm{s}}\text{ is faulty}\\0\text{ otherwise}\end{cases},\quad \forall n_{\mathrm{v}}\in N_{\mathrm{V}},\ \forall n_{\mathrm{s}}\in N_{\mathrm{S}}, \tag{10}$$

$$s(e_{\mathrm{v}})=\begin{cases}1\ iff\ e_{\mathrm{v}}\text{ is faulty}\\0\text{ otherwise}\end{cases},\ s(e_{\mathrm{s}})=\begin{cases}1\ iff\ e_{\mathrm{s}}\text{ is faulty}\\0\text{ otherwise}\end{cases},\quad \forall e_{\mathrm{v}}\in E_{\mathrm{V}},\ \forall e_{\mathrm{s}}\in E_{\mathrm{S}}, \tag{11}$$

$$s(\boldsymbol{G}_{\mathrm{V}})=\begin{cases}1\ iff\ \boldsymbol{G}_{\mathrm{V}}\text{ is faulty}\\0\text{ otherwise}\end{cases},\ s(\boldsymbol{G}_{\mathrm{S}})=\begin{cases}1\ iff\ \boldsymbol{G}_{\mathrm{S}}\text{ is faulty}\\0\text{ otherwise}\end{cases}.$$

In constraint (8), if virtual node n_{v} is re-embedded onto substrate node $\mathrm{n_s}$, $x(n_{\mathrm{v}}, n_{\mathrm{s}}) = 1$. Otherwise, $x(n_{\mathrm{v}}, n_{\mathrm{s}}) = 0$. In constraint (9), if virtual link e_{up} is re-embedded onto substrate link e_{ij}, $x(e_{up}, e_{ij}) = 1$. Otherwise, $x(e_{up}, e_{ij}) = 0$. In constraint (10), if virtual node n_{v} is faulty, $s(n_{\mathrm{v}}) = 1$. Otherwise, $s(n_{\mathrm{v}}) = 0$. Also, if substrate node n_{s} is faulty, $s(n_{\mathrm{s}}) = 1$. Otherwise, $s(n_{\mathrm{s}}) = 0$. In constraint (11), if virtual link e_{v} is faulty, $s(e_{\mathrm{v}}) = 1$. Otherwise, $s(e_{\mathrm{v}}) = 0$. If substrate link e_{s} is faulty, $s(e_{\mathrm{s}}) = 1$. Otherwise, $s(e_{\mathrm{s}}) = 0$. If virtual network $\boldsymbol{G}_{\mathrm{V}}$ is faulty, $s(\boldsymbol{G}_{\mathrm{V}}) = 1$. Otherwise, $s(\boldsymbol{G}_{\mathrm{V}}) = 0$. If substrate network $\boldsymbol{G}_{\mathrm{S}}$ is faulty, $s(\boldsymbol{G}_{\mathrm{S}}) = 1$. Otherwise, $s(\boldsymbol{G}_{\mathrm{S}}) = 0$.

$$\forall n_{\mathrm{v}}\in N_{\mathrm{V}},\ \forall n_{\mathrm{s}}\in N_{\mathrm{S}}, cpu(n_{\mathrm{v}})\times x(n_{\mathrm{v}},n_{\mathrm{s}})\le cpu(n_{\mathrm{s}}), \tag{12}$$

$$\forall e_{up}\in E_{\mathrm{V}},\ \forall e_{ij}\in E_{\mathrm{S}}, \sum_{\forall e_{up}\in E_{\mathrm{V}}} bw(e_{up})\times x(e_{up},e_{ij})\le bw(e_{ij}). \tag{13}$$

Constrains (12) and (13) are resource constraints, and they ensure that if virtual node n_{v} and virtual link e_{up} are selected to re-embed, the candidate substrate node n_{s} and substrate link e_{ij} should have more resources than the virtual node n_{v} and virtual link e_{up} request, respectively.

$$x(n_{\mathrm{v}},n_{\mathrm{s}})\times dis(loc(n_{\mathrm{s}}),loc(n_{\mathrm{v}}))\le D(n_{\mathrm{s}}). \tag{14}$$

Constrain (14) is location constraint and $dis(loc(n_s), loc(n_v))$ denotes the distance between substrate node n_s and virtual node n_v.

$$\forall n_v \in N_V,\ \forall n_s \in N_S,\ \forall e_{up} \in E_V,\ \forall e_{ij} \in E_S,$$

$$\sum_{e_{ji} \in E_S} x(e_{up}, e_{ji}) - \sum_{e_{ij} \in E_S} x(e_{up}, e_{ij}) = \begin{cases} 1, & x(n_u, n_j) = 1 \\ -1, & x(n_u, n_j) = 1 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

Constrain (15) is the connectivity constraint. It refers to the flow conservation constraint for routing one unit of net flow from virtual node n_u to virtual node n_p.

$$\forall n_s \in N_S,\ \sum_{n_v \in N_V} x(n_v, n_s) \leq 1 \tag{16}$$

$$\forall n_v \in N_V,\ \sum_{n_s \in N_S} x(n_v, n_s) = 1 \tag{17}$$

Constrains (16) and (17) are re-embedding constraints. Constrain (16) ensures that all virtual nodes in same VN are re-embedded on different substrate nodes. Constrain (17) ensures that each virtual node is re-embedded on up to one substrate node.

$$\text{if } \forall n_v \in N_V \text{ and } s(n_v) = 1, \text{ then } s(\boldsymbol{G}_V) = 1 \tag{18}$$

$$\text{if } \forall e_{up} \in E_V \text{ and } s(e_{up}) = 1, \text{ then } s(\boldsymbol{G}_V) = 1 \tag{19}$$

$$\forall n_i \in N_S,\ \forall n_j \in N_S,\ \forall e_{ij} \in E_S, \text{ if } s(n_i) = 1, \text{ then } s(e_{ij}) = 1 \tag{20}$$

$$\forall n_q \in N_{Vf},\ \forall n_j \in N_{Sf},\ x(n_q, n_j) = 0 \tag{21}$$

$$\forall e_{up} \in E_{Vf},\ \forall e_{ij} \in E_{Sf},\ x(e_{up}, e_{ij}) = 0 \tag{22}$$

Constrains (18)–(22) are failure constraints. Constrains (18) and (19) ensure that if either the virtual node or virtual link fail in VN, the VN will also fail. Constrain (20) ensures that if one of the end-nodes in substrate link fails, the substrate link will fail. Constrain (21) ensures that virtual node and the corresponding substrate node re-embedded by it cannot fail at the same time. Constrain (22) ensures that virtual link and the corresponding substrate link re-embedded by it cannot fail at the same time.

This ILP is known to be NP-hard problem, and solving it is computationally intractable. Even though optimal results can be obtained by some exact algorithms or open source linear programming toolkit GLPK, they may incur exponential increasing running time. Consequently, they cannot be scaled to address large-scale VNs. Hence, most researchers solve the ILP problem of VN reconfiguration by proposing a corresponding heuristic algorithm which has short computational time and gets an approximate optimal solution. Therefore, we propose a novel heuristic algorithm called DVNRM-HMF-WRE to solve the ILP of VN reconfiguration.

4. DVNRM-HMF-WRE

4.1. Problem Statement

The VN reconfiguration is a complex and dynamic process. There are many event statements at the same time. We analyze the possible event statements and provide the dynamic process of VN reconfiguration first. Then, three algorithms in DVNRM-HMF-WRE are proposed.

VN embeds and leaves dynamically. Many types of failures occur randomly in SN. At time t, there are four possible event statements:

1. When the lifetime of VN ends, the corresponding SN resources are released.
2. When the faulty component in SN is repaired, its resources also return to normal.
3. When SN failure occurs, the resources of faulty component are set to zero and the VN embedded on it needs to be re-embedded. If multiple failures occur at the same time, the faulty VNs will be re-embedded one by one. When all faulty virtual components in one VN are re-embedded successfully, the VN reconfiguration is successful.
4. When VN request arriving, VN begins to embed. If succeed, the corresponding resources are used.

When multiple SN failures occur, the process of DVNRM-HMF-WRE is shown in Figure 2.

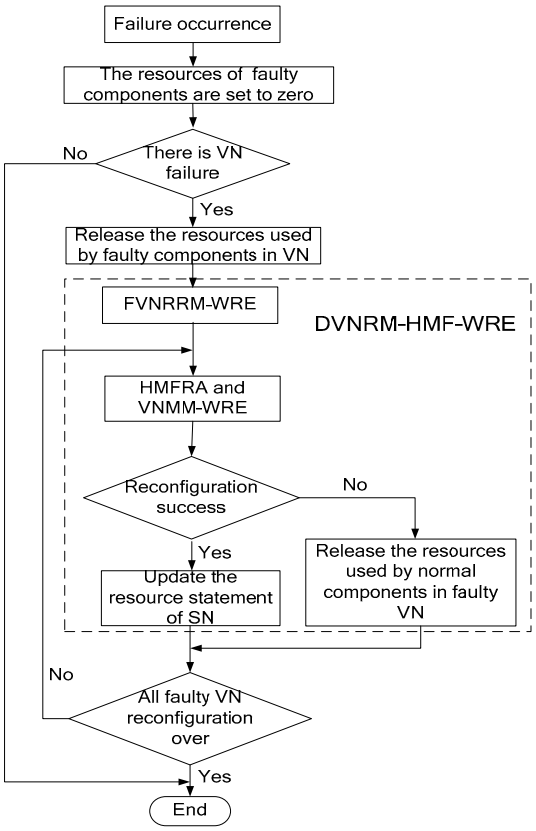

Figure 2. The process of DVNRM-HMF-WRE.

As can be seen that, from Figure 2, when failure occurs, the resources of faulty components are set to zero. If there are some VNs embedded onto the faulty components, the resources used by faulty components in VNs are released. Then, the DVNRM-HMF-WRE is used to re-embed the faulty VNs. In DVNRM-HMF-WRE, the FVNRRM-WRE is used to rank multiple faulty VNs to decide which faulty VN should be re-embedded first. The HMFRA and VNMM-WRE are used together to recover the hybrid multiple failures in one VN. The HMFRA is proposed to rank hybrid multiple faulty nodes/links in one VN, and the VNMM-WRE is provided to improve the efficiency of virtual node migration. If the VN reconfiguration is successful, the resource statement of SN I updated, otherwise, the resources used by normal components in faulty VN are released. If all VN reconfigurations are over, the DVNRM-HMF-WRE is stopped.

4.2. FVNRRM-WRE

When multiple failures occur at the same time, and substrate resources are limited, the reconfiguration sequence of faulty VNs is especially important. In faulty VN ranking strategies, ranking indicators and ranking method are very important. Among them, faulty component number, VN revenue, and remaining lifetime of the faulty VN all have obvious impact on the reconfiguration performance. They are selected to rank the faulty VNs. When ranking faulty VNs, a fixed formula is usually used, and it cannot adapt to the changing environment. Therefore, the WRE method is introduced in this paper.

The WRE method is a classical ranking method which can adjust the indicator coefficients to adapt the changing environment. It is improved by the classical technique for order preference by similarity to an ideal solution (TOPSIS) [20]. In TOPSIS, the generalized distance is unable to distinguish the point in perpendicular bisector between positive and negative ideal solution [21]. Therefore, the relative entropy is used, instead, to the generalized distance.

For system A and B in state A_i and B_i (i = 1, 2, ... ,N), their relative entropy, C, can be defined by

$$C = \sum_{i=1}^{N} [A_i \log \frac{A_i}{B_i} + (1 - A_i) \log \frac{1 - A_i}{1 - B_i}]. \tag{23}$$

There are N faulty VNs, and each faulty VN has M evaluation indicators. The j-th indicator coefficient of the i-th node is denoted as X_{ij} (i = 1, 2, ... , N; j = 1, 2, ... , M), and all coefficients of faulty VNs constitute a decision matrix $\boldsymbol{X}$ which is denoted as

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ x_{21} & x_{22} & \cdots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NM} \end{pmatrix}. \tag{24}$$

Also, the coefficient is normalized to make a fair comparison between different indicators.

In different environments, each indicator has a different importance. The weighting coefficient of the j-th indicator is expressed as $\omega_j (j = 1, 2, \ldots, M, \sum \omega_j = 1)$, and the weighted normalized decision matrix is denoted by

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\omega} = \begin{pmatrix} x_{11}\omega_1 & x_{12}\omega_2 & \cdots & x_{1M}\omega_M \\ x_{21}\omega_1 & x_{22}\omega_2 & \cdots & x_{2M}\omega_M \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1}\omega_1 & x_{N2}\omega_2 & \cdots & x_{NM}\omega_M \end{pmatrix}. \tag{25}$$

The positive and negative ideal solutions A^+ and A^- are determined as

$$A^+ = \{\max(y_{i1}, y_{i2}, \ldots, y_{iM})\} = \{y_1^{\max}, y_2^{\max}, \ldots, y_M^{\max}\}, \tag{26}$$

$$A^- = \{\min(y_{i1}, y_{i2}, \ldots, y_{iM})\} = \left\{y_1^{\min}, y_2^{\min}, \ldots, y_M^{\min}\right\}. \tag{27}$$

The relative entropies of each solution to positive and negative ideal solution are calculated as

$$C_i^+ = \sum_{j=1}^{M} \left[y_j^{\max} \log \frac{y_j^{\max}}{y_{ij}} + (1 - y_j^{\max}) \log \frac{1 - y_j^{\max}}{1 - y_{ij}} \right], \tag{28}$$

$$C_i^- = \sum_{j=1}^{M} \left[y_j^{\min} \log \frac{y_j^{\min}}{y_{ij}} + (1 - y_j^{\min}) \log \frac{1 - y_j^{\min}}{1 - y_{ij}} \right]. \tag{29}$$

The similarity between each solution and the ideal one is calculated as

$$Z_i = \frac{C_i^-}{C_i^- + C_i^+}, 0 \leq Z_i \leq 1. \tag{30}$$

The details of FVNRRM-WRE method are described in Algorithm 1.

Algorithm 1. The FVNRRM-WRE

Input: $\boldsymbol{X}$, weighting coefficients
Output: Z_i
1. Bring $\boldsymbol{X}$ and the weighting coefficient of each indicator into Formula (25) to construct the weighted normalized matrix $\boldsymbol{Y}$.
2. Calculate the positive and negative ideal scheme A^+ and A^-.
3. Calculate the positive and negative ideal solution C^+ and C^-.
4. Calculate Z_i and get the importance degrees of faulty VNs.

In Algorithm 1, $\boldsymbol{X}$ is the multiple indicators decision matrix which includes three indicators, called faulty components number, VN revenue, and remaining lifetime of the faulty VN. To calculate Z_i, which is the importance degree of faulty VN_i, the WRE method is widely used based on Formulas (26)–(30). After getting the importance degrees of faulty VNs, rank the faulty VNs in order of importance degree, from large to small, which can re-embed the important VN first.

4.3. HMFRA

In a faulty VN, there are multiple failures which include node failures and link failures. Recovery sequence is important in VN reconfiguration, and the HMFRA is proposed to recover different types of failures in a suitable manner. The details of HMFRA are shown in Algorithm 2.

As can be seen from Algorithm 2:

1. Faulty virtual node and its connective links (VN link failures I) are recovered first. If substrate node is faulty, virtual node and its connective virtual links both need to be re-embedded. They will consume huge substrate node and link resources. They are the hardest to re-embed. If this succeeds, go on re-embedding virtual links in the set of VN link failure II. Otherwise, VN reconfiguration has failed.
2. If the VN link failures II are re-embedded successfully, then judge if there are VN link failures III. If there are VN link failures III, re-embed them. If this succeeds, VN reconfiguration is successful.
3. If there is no VN link failure III after re-embedding the VN link failure II successfully, the faulty VN returns to normal. If the VN link failure II fails to re-embed, the faulty VN reconfiguration has failed.
4. If there is only VN link failure III, re-embed these faulty virtual links. If this succeeds, VN reconfiguration is successful.

Algorithm 2. The HMFRA

Input: G_S, G_V, the set of faulty node, the set of VN link failure I, the set of VN link failure II, the set of VN link failure III
Output: G_S, G_V

```
if the faulty node set is not empty
    Migrate the virtual nodes using VNMM-WRE and re-embed all their connective links (VN
    link failures I) using k-shortest path algorithm.
    if all virtual nodes are migrated and all their adjacent links are re-embedded successfully
      Update G_S and G_V
      if the set of VN link failure II is not empty
          Re-embed all VN link failures II
        if all VN link failures II are re-embedded successfully
          Update G_S and G_V
          if the set of VN link failure III is not empty
            Re-embed all of them
            if all VN link failures III are re-embedded successfully
              Update G_S and G_V
              Return VN_RE-EMBED_SUCCESS
            else release the resources in G_S which are used by the normal parts
              Update G_S
              Return VN_RE-EMBED_FAILURE
            end if
            else update G_S and G_V
              Return VN_RE-EMBED_SUCCESS
            end if
        else release the resources in G_S which are used by the normal parts
            Update G_S
            Return VN_RE-EMBED_ FAILURE
        end if
      else update G_S and G_V
            Return VN_RE-EMBED_SUCCESS
      end if
    else release the resources in G_S which are used by the normal parts
            Update G_S
            Return VN_RE-EMBED_ FAILURE
    end if
  else
   if the set of VN link failure III is not empty
        Re-embed all of them
        if all VN link failures III are re-embedded successfully
          Update G_S and G_V
          Return VN_RE-EMBED_SUCCESS
        else release the resources in G_S which are caused by the normal parts
            Update G_S
            Return VN_RE-EMBED_ FAILURE
        end if
    end if
  end if
```

4.4. VNMM-WRE

In HMFRA, the method of faulty virtual node migration is called VNMM-WRE, which is proposed in this section and the method of virtual link re-embedding is *k*-shortest path algorithm [22].

Similar to FVNRRM-WRE, the WRE method is also introduced into VNMM-WRE. In virtual node ranking, the importance degree of each virtual node $Z(n_v)$ is calculated using WRE method, which selects node CPU resource and node adjacent link bandwidth resource as the indicators. In substrate node ranking, the importance degree of each substrate node $Z(n_s)$ is calculated using WRE method which selects node CPU resource, node adjacent link bandwidth resource and the reciprocal of hop counts between candidate substrate node and substrate node which is embedded by the neighboring virtual node of the migrated virtual node as the indicators [23].

The details of VNMM-WRE are shown in Algorithm 3

Algorithm 3. The VNMM-WRE

Input: Faulty virtual nodes which need to migrate
Output: NodeMigraingList
1. for each virtual node $n_v \in N_V$
2. Calculate the $Z(n_v)$
3. end for
4. Rank virtual nodes in order of $Z(n_v)$ from large to small
5. Save virtual node ranking order into *VirtualNodeList*
6. for each virtual node in *VirtualNodeList*
7. Select the candidate substrate node set $Can(n_{vi})$ which satisfies the resource constraints
8. Remove the embedded substrate nodes from $Can(n_{vi})$
9. if $Can(n_{vi})$ is empty
10. Return NODE_ MIGRATE_FAILURE
11. else
12. for each candidate node n_s in $Can(n_{vi})$
13. Calculate the $Z(n_s)$ of each substrate node in $Can(n_{vi})$
14. end for
15. for each virtual node in $Can(n_{vi})$
16. Re-embed n_v to n_s which has the largest $Z(n_s)$ value
17. Re-embed the virtual links which connect the virtual node
18. if all virtual links are re-embedded successfully
19. Save the substrate node n_s into *NodeMigratingList*
20. break
21. else
22. Remove the virtual node n_v from $Can(n_{vi})$
23. end if
24. end for
25. Update $\boldsymbol{G}_S$ and $\boldsymbol{G}_V$
26. Return NODE_MIGRATE_SUCCESS
27. end if
28. end for

In Algorithm 3:

1. Lines 1–5: calculate the $Z(n_v)$ of each faulty virtual node and rank them based on $Z(n_v)$. Then save the ranking results into *VirtualNodeList*.
2. Lines 6–14: select the candidate substrate nodes and calculate their $Z(n_s)$.
3. Lines 15–28: re-embed the virtual node and its adjacent virtual links.

In summary, with the help of FVNRRM-WRE, HMFRA, and VNMM-WRE, the faulty VNs caused by hybrid multiple failures can be recovered.

4.5. COMPLEXITY ANALYSIS

The DVNRM-HMF-WRE includes FVNRRM-WRE, HMFRA, and VNMM-WRE. In FVNRRM-WRE, all faulty VNs are ranked based on the WRE method. Its complexity is $O(|N_{FN}|)$, in which $|N_{FN}|$ is the total number of faulty VNs. The HMFRA and VNMM-WRE are used, together, to recover the hybrid multiple failures in VNs. In the recovery of faulty VNs, the complexity of virtual node re-embedding is $O(|N_V||N_S|^2)$. $|N_V|$ is the total number of faulty virtual nodes in all faulty VNs and $|N_S|$ is the total number of substrate nodes. The virtual link re-embedding algorithm is k-shortest path algorithm and its complexity is $O(k(|N_V| + |E_V|)(|E_S| + |N_S|\lg|N_S|))$. $|E_V|$ represents the total number of faulty virtual links in all faulty VNs and $|E_S|$ represents the total number of substrate links. Therefore, the total complexity of DVNRM-HMF-WRE is $O(|N_{FN}| + |N_V||N_S|^2 + k(|N_V| + |E_V|)(|E_S| + |N_S|\lg|N_S|))$.

5. Simulation

In this section, we present the performance evaluation of the proposed DVNRM-HMF-WRE.

5.1. Simulation Setup

We run our experiments on a workstation with Lenovo Tianyi 510Pro (Lenovo, Beijing, China) with Windows 10 system (Microsoft Corporation, Redmond, WA, USA). The hardware platform is the Inter Core i7-7700 3.6 GHz process (Intel Corporation, Santa Clara, CA, USA) with 8 GB RAM (Intel Corporation, Santa Clara, CA, USA). The software is MATLAB R2007a (MathWorks, Natick, MA, USA). We run our simulations for 3000 time units to leave the performance in a stable state. Each simulation is performed 10 times, and we take the average values as the final results. Simulation codes and results could be found in the Supplementary File.

The SN topology and VN topology used in simulation are generated by the improved Salam network topology random generation algorithm [24]. It is worth noting that, the main parameters of VN, SN, and failures are summarized in Table 2. In Table 2, [*xmin*, *xmax*] denotes a uniform distribution between *xmin* and *xmax*. *Pois*{*p*} and *Expo*{*g*} stand for the Poisson and exponential distributions, with mean p and g, respectively.

Table 2. Simulation parameters.

Characteristics	Values	Characteristics	Values
Number of substrate node	100	Number of virtual node per VN	[5, 10]
Substrate node capacity	[50, 100]	Virtual node capacity	[2, 10]
Number of substrate link	500	Virtual link connection probability	0.5
Substrate link capacity	[50, 100]	Virtual link capacity	[2, 10]
VN arrive rate	*Pois*{0.1}	VN lifetime	*Expo*{300}
Failure arrive rate	*Pois*{0.2}	Failure lifetime	*Expo*{500}
Network distance scope	1000 × 1000	Virtual node position constrain	400

5.2. Scenarios

Five scenarios are set in our simulation to validate the performance of DVNRM-HMF-WRE.

In the first scenario, the performance of DVNRM-HMF-WRE is compared with two typical VN reconfiguration methods TA-VNR [15] and ReNoVatE [18].

In the second scenario, our FVNRRM-WRE is compared with the other three faulty VN ranking methods to evaluate the impact of the FVNRRM-WRE. The details of them are shown in Table 3.

Table 3. Comparison of faulty VN ranking methods.

Algorithm	Description
DVNRM-HMF-WRE	VN reconfiguration method proposed in this paper which ranks faulty VNs based on FVNRRM-WRE.
DVNRM-HMF-FN	VN reconfiguration method which ranks faulty VNs based on faulty component number.
DVNRM-HMF-VNR	VN reconfiguration method which ranks faulty VNs based on VN revenue.
DVNRM-HMF-RLT	VN reconfiguration method which ranks faulty VNs based on remaining lifetime of the faulty VN.

In the third scenario, our HMFRA is compared with the other two failure handling methods to evaluate the impact of the HMFRA. The details of them are shown in Table 4.

Table 4. Comparison of failure handling methods.

Algorithm	Description
DVNRM-HMF-WRE	VN reconfiguration method proposed in this paper which handles different failures based on HMFRA.
DVNRM-HMF-WRE II	VN reconfiguration method which handles VN link failure II first.
DVNRM-HMF-WRE III	VN reconfiguration method which handles VN link failure III first.

In the fourth scenario, our VNMM-WRE is compared with the other two VN node migration methods which include DVNRM-HMF-HOPS and DVNRM-HMF-RES. The details of them are shown in Table 5.

Table 5. Comparison of VN node migration methods.

Algorithm	Description
DVNRM-HMF-WRE	VN reconfiguration method proposed in this paper, in which virtual node is migrated based on VNMM-WRE.
DVNRM-HMF-RES	VN reconfiguration method in which virtual node is migrated based on the total node and link resources.
DVNRM-HMF-HOPS	VN reconfiguration method in which virtual node is migrated based on the reciprocal of hop counts between candidate substrate node and substrate node which is embedded by the neighboring virtual node of the migrated virtual node.

At last, the arrival rate of failure, u, and the lifetime of failure, T, are set to evaluate their effects on DVNRM-HMF-WRE.

5.3. Simulation Results

5.3.1. Comparison of Different VN Reconfiguration Methods

The results of comparison of faulty VN reconfiguration methods are shown in Figure 3.

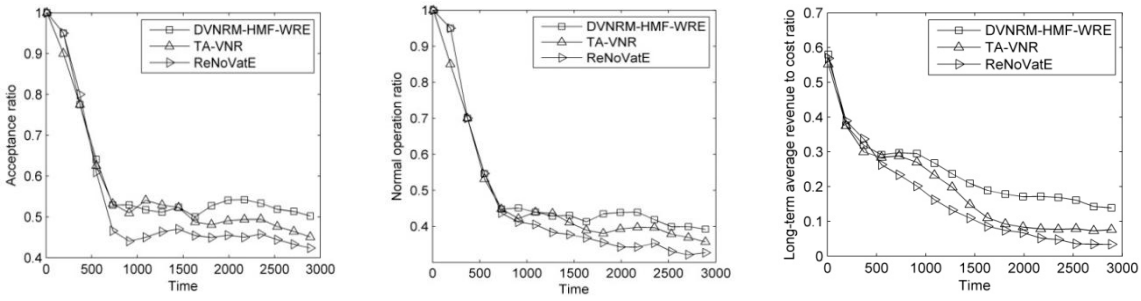

Figure 3. Comparison of DVNRM-HMF-WRE, TA-VNR and ReNoVatE. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 3a–c, the DVNRM-HMF-WRE proposed in this paper has the best AR, NOR, and LAR/CR. It achieves 5.7% and 11.3% higher AR than TA-VNR and ReNoVatE, respectively. Its NOR is 6.9% and 11% higher than TA-VNR and ReNoVatE, respectively. Its LAR/CR is 25.7% and 41.9% higher than TA-VNR and ReNoVatE, respectively. In DVNRM-HMF-WRE, faulty components number, VN revenue, and remaining lifetime of faulty VN are all considered to rank the faulty VNs. The resource and topology indicators are both used based on WRE in VNMM-WRE. In addition, the faulty virtual node and VN link failures I are recovered first. Therefore, the performance of DVNRM-HMF-WRE is the best.

5.3.2. Comparison of Faulty VN Ranking Methods

The results of comparison of faulty VN ranking methods are shown in Figure 4.

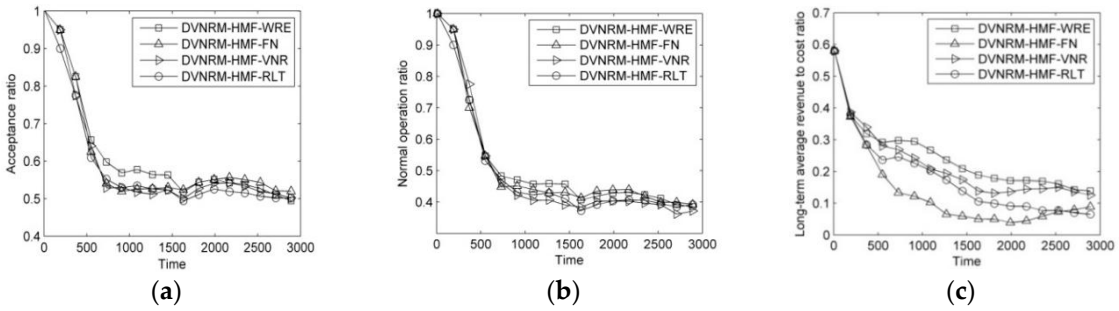

Figure 4. Comparison of different faulty VN ranking methods. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 4a–c, the DVNRM-HMF-WRE achieves 4.5%, 5.3%, and 5.6% higher AR than DVNRM-HMF-FN, DVNRM-HMF-VNR, and DVNRM-HMF-RLT, respectively. The DVNRM-HMF-WRE achieves 3.1%, 6.4%, and 5.1% higher NOR than DVNRM-HMF-FN, DVNRM-HMF-VNR, and DVNRM-HMF-RLT, respectively. The DVNRM-HMF-WRE achieves 31.1%, 17.4%, and 25.1% higher LAR/CR than DVNRM-HMF-FN, DVNRM-HMF-VNR, and DVNRM-HMF-RLT, respectively. In this paper, our objective is to maximize the LAR/CR. If re-embedding the faulty VN fails, a high penalty cost will be paid. Improving the proportion of VN revenue, remaining lifetime of the faulty VN, and re-embedding the faulty VN which has higher penalty cost and revenue based on WRE, can improve the LAR/CR significantly.

5.3.3. Comparison of Failure Handling Methods

The results of comparison of failure handling methods are shown in Figure 5.

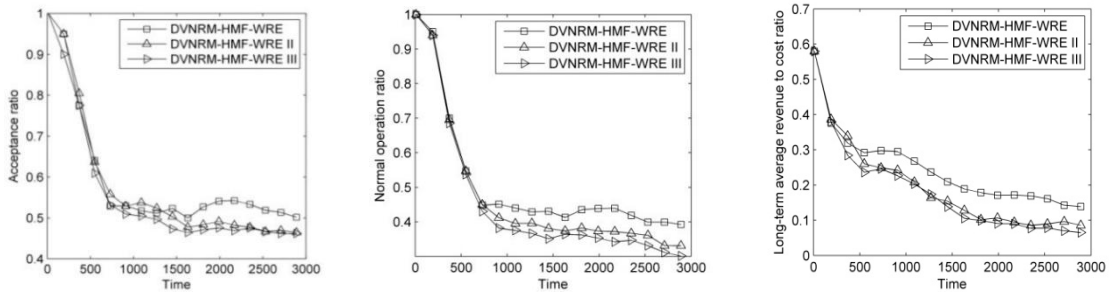

Figure 5. Comparison of different failure handling methods. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 5a–c, the performance of DVNRM-HMF-WRE is better than the other two failure handling methods. It achieves 5.7% and 7.3% higher AR than DVNRM-HMF-WRE II and DVNRM-HMF-WRE III, respectively. Its NOR is 8.5% and 12.8% higher than DVNRM-HMF-WRE II and DVNRM-HMF-WRE III, respectively. Its LAR/CR is 24% and 27.9% higher than DVNRM-HMF-WRE II and DVNRM-HMF-WRE III, respectively. When the substrate node is faulty, the virtual node and its connective virtual links (VN link failure I) will fail at the same time. Substrate node and link resources are both consumed to re-embed the faulty VN. Compared with the other two link failures, node failure is the hardest to recover. Therefore, re-embedding faulty virtual node and its connective virtual links, first, can improve the performance of VN recovery algorithm.

5.3.4. Comparison of Different VN Node Migration Methods

The results of comparison of VN node migration methods are shown in Figure 6.

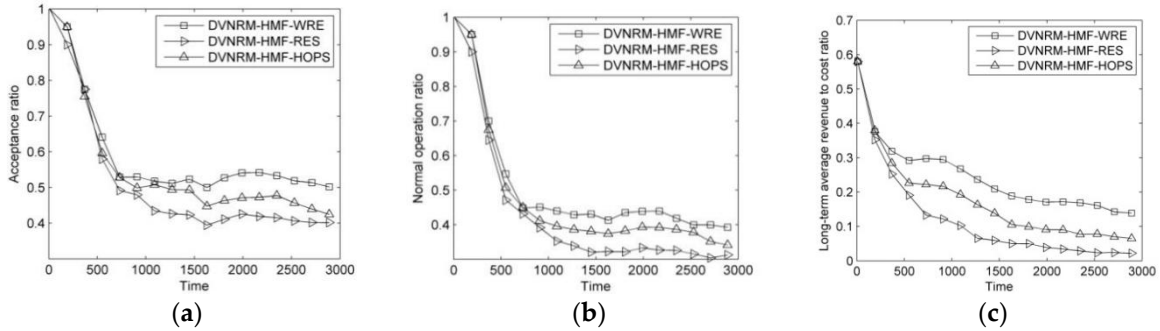

Figure 6. Comparison of different VN node migration methods. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 6a,b, DVNRM-HMF-WRE has the best AR and NOR. It achieves approximately 8.3% and 16.3% higher AR than VNMM-HMF-HOPS and VNMM-HMF-RES, respectively. The NOR of VNMM-HMF-WRE is approximately 7.4% and 17.4% higher than VNMM-HMF-HOPS and VNMM-HMF-RES, respectively. In Figure 6c, the advantage of VNMM-WRE is more obvious in LAR/CR. It is approximately 31.8% and 68.3% higher than VNMM-HMF-HOPS and VNMM-HMF-RES, respectively. In VNMM-HMF-WRE, the resource and topology indicators are both used to reduce substrate resource consumption in VN reconfiguration. Also, WRE is introduced to calculate the candidate substrate node importance.

5.3.5. The Effect of Arrival Rate of Failure and Lifetime of Failure on DVNRM-HMF-WRE

In this section, the arrival rate of failure u and the lifetime of failure T are selected to describe their effects on DVNRM-HMF-WRE.

(1) The Arrival Rate of Failure u

In this section, the arrival rates of failure are set to 1/3, 1/6, and 1/9. The results of DVNRM-HMF-WRE in different arrival rates of failure are shown in Figure 7.

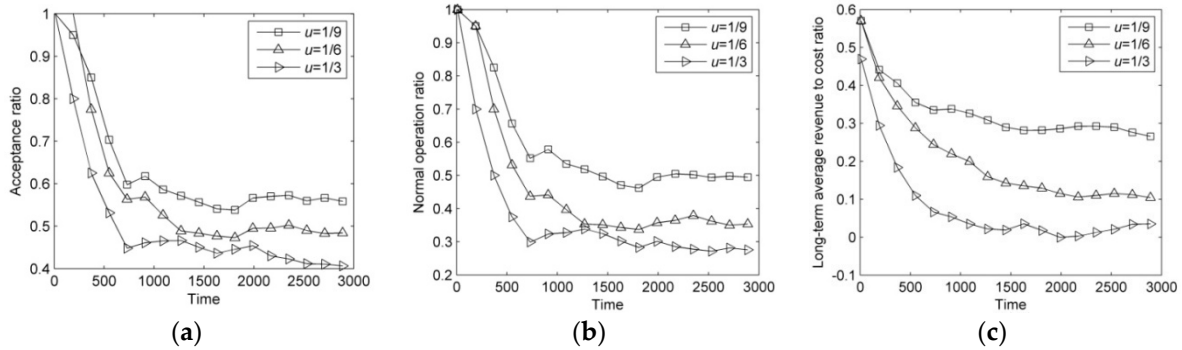

Figure 7. Comparison of DVNRM-HMF-WRE in different arrival rates of failure. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 7a–c, with the increase of arrival rate of failure u, the number of SN failures keeps increasing, which results in a reduction in resources used for VN embedding and reconfiguration. Therefore, the AR and NOR of VN are decreasing. A large number of penalty cost caused by the failure of VN reconfiguration significantly reduces the LAR/CR.

(2) The Lifetime of Failure T

In this section, the lifetimes of failure are set to 400, 600, and 800. The results of DVNRM-HMF-WRE in different lifetimes of failure are shown in Figure 8.

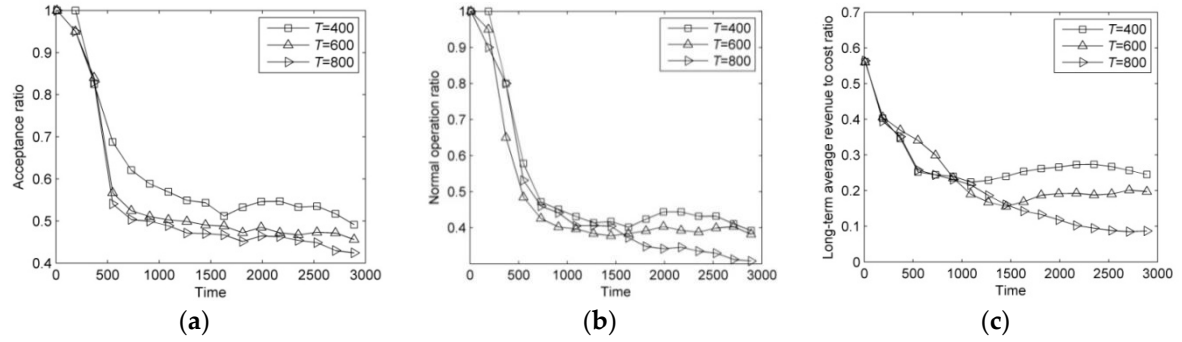

Figure 8. Comparison of DVNRM-HMF-WRE performance in different arrival rates of failure. (**a**) Acceptance ratio; (**b**) Normal operation ratio; (**c**) Long-term average revenue to cost ratio.

As can be seen from Figure 8a–c, increasing the lifetime of failure can reduce the available substrate resources, which not only decreases the AR, but also results in a reduction in resources used for VN reconfiguration. Therefore, with the increase of the lifetime of failure, the AR, NOR, and LAR/CR are all reduced significantly.

6. Conclusions and Future Work

In this paper, we have addressed the problem of recovering faulty VNs affected by hybrid multiple failures in SN. In this regard, we have formulated the VN reconfiguration to an ILP model to maximize the LAR/CR. We have proposed an efficient heuristic policy called DVNRM-HMF-WRE to solve this ILP model. In DVNRM-HMF-WRE, the WRE method is introduced into FVNRRM-WRE and VNMM-WRE. In FVNRRM-WRE, the WRE method is used to rank the multiple faulty VNs. In VNMM-WRE, candidate substrate node are selected based on WRE method which takes resource

and topology indicators into consideration. Also, a failure handling method called HMFRA is proposed, which handles virtual node failure and VN link failure I first. At last, five experiments are designed. The first experiment verifies that the proposed DVNRM-HMF-WRE has excellent performance than other typical VN reconfiguration methods. The next three experiments assess that our proposed faulty VN ranking method, failure handling method, and VN node migration method perform better than other corresponding methods. The last experiment sets two different scenarios to evaluate the performance of DVNRM-HMF-WRE in different arrival rates and lifetimes of failures. Evaluation results show that decreasing the arrival rate and the lifetime of failure can improve the performance of VN reconfiguration.

In future research, we will introduce the privacy and energy consumption into VN reconfiguration. Without adequate protection, users from a VN might be able to gain unauthorized access to data being transmitted through other VNs, violating the privacy of the entities that own those networks. Hence, privacy-oriented VNE and preserving privacy with VN stacks gradually get more and more attention. Also, energy consumption has become another hot topic in NV. With the increase of communication traffic every year, some energy models have been proposed to minimize the energy consumption in NV, such as the load-dependent power consumption model in VNE, the energy-efficient resource management for real-time service function chain in fog-supported SDN and network-aware energy optimization model for VN. Therefore, the ILP formulation and the heuristic algorithm could be extended to consider the privacy and energy consumption in future work.

Supplementary Materials: supplementary materials are available online at http://www.mdpi.com/1099-4300/20/9/711/s1.

Author Contributions: Y.S. performed initially the theoretical derivation, designed the experiments and software code, and wrote the draft of this paper in Chinese; X.M. proposed the original idea, checked the initial derivation, revised radically the paper, and rewrote the paper in English; Q.K. made a major contribution to the revision of this paper as for the reviews' comments. X.H. analyzed the experimental data.

Funding: This research was funded by [National Science Foundation of China] grant number [61401499].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NV	network virtualization
VN	virtual network
SN	substrate network
SDN	software defined network
ILP	integer linear programming
DVNRM-HMF-WRE	dynamic virtual network reconfiguration method for hybrid multiple failures based on weighted relative entropy
FVNRRM-WRE	faulty VN reconfiguration ranking method based on weighted relative entropy
HMFRA	hybrid multiple failures ranking algorithm
VNMM-WRE	virtual node migration method based on weighted relative entropy
WRE	weighted relative entropy
AR	acceptance ratio
NOR	normal operation ratio
LAR/CR	long-term average revenue to cost ratio

References

1. Fischer, A.; Botero, J.F.; Beck, M.T.; de Meer, H.; Hesselbach, X. Virtual network embedding: A survey. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 1888–1906. [CrossRef]
2. Zhang, J.; Zhao, C.; Wu, H.; Lin, M.; Duan, R. Virtual network embedding based on graph entropy. *Entropy* **2016**, *20*, 315. [CrossRef]
3. Mijumbi, R.; Serrat, J.; Gorricho, J.L.; Bouten, N.; Turck, F.D.; Boutaba, R. Network function virtualization: State-of-the-art and research challenges. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 236–262. [CrossRef]

4. Tajiki, M.M.; Shojafar, M.; Akbari, B.; Salsano, S.; Conti, M.; Singhal, M. Joint failure recovery, fault prevention, and energy-efficient resource management for real-time SFC in fog-supported SDN. *Comput. Netw.* Available online: http://cn.arxiv.org/pdf/1807.00324 (accessed on 1 July 2018).
5. Shojafar, M.; Canali, C.; Lancellotti, R.; Baccarelli, E. Minimizing computing-plus-communication energy consumptions in virtualized networked data centers. In Proceedings of the IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 27–30 June 2016; pp. 94–104.
6. Shahriar, N.; Chowdhury, S.R.; Ahmed, R. Virtual network survivability through joint space capacity allocation and embedding. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 502–518. [CrossRef]
7. Boem, F.; Ferrari, R.M.G.; Keliris, C.; Parisini, T.; Polycarpou, M. A distributed networked approach for fault detection of large-scale systems. *IEEE Trans. Autom. Control* **2016**, *62*, 18–33. [CrossRef]
8. Herker, S.; Khan, A.; An, X. Survey on survivable virtual network embedding problem and solutions. In Proceedings of the 9th International Conference on Networking and Services (ICNS), Lisbon, Portugal, 24–29 March 2013; pp. 94–104.
9. Li, R.; Wu, Q.; Tan, Y.; Zhang, J. On the optimal approach of survivable virtual network embedding in virtualized SDN. *IEICE Trans. Inf. Syst.* **2018**, *101*, 698–708. [CrossRef]
10. Sangjin, H.; Jason, P.J.; Pyungkoo, P.; Hosun, Y.; Hoyong, R.; Sungback, R. Survivable virtual topology design in multi-domain optical networks. *J. Opt. Commun. Netw.* **2016**, *8*, 408–416. [CrossRef]
11. Aguado, A.; Davis, M.; Peng, S.; Alvarez, M.V.; Lopez, V.; Szyrkowiec, T.; Autenrieth, A.; Vilalta, R.; Mayoral, A.; Muñoz, R.; et al. Dynamic virtual network reconfiguration over SDN orchestrated multi-technology optical transport domains. *J. Lightw. Technol.* **2016**, *34*, 1933–1938. [CrossRef]
12. Chowdhury, S.R.; Ahmed, R.; Khan, M.M.A.; Shahriar, N.; Boutaba, R.; Mitra, J.; Zeng, F. Dedicated protection for survivable virtual network embedding. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 913–926. [CrossRef]
13. Ayoubi, S.; Chen, Y.; Assi, C. Towards promoting backup-sharing in survivable virtual network design. *IEEE/ACM Trans. Netw.* **2016**, *24*, 3218–3231. [CrossRef]
14. Chang, X.; Muppala, J.K.; Wang, B.; Liu, J.; Sun, L. Migration cost aware virtual network re-embedding in presence of resource failures. In Proceedings of the 18th IEEE International Conference on Networks (ICON), Singapore, 12–14 December 2012; pp. 24–29.
15. Peng, L. A topology-awareness virtual network reconfiguration algorithm. *J. Sichuan Univ.* **2015**, *47*, 110–115. [CrossRef]
16. Ghaleb, A.M.; Khalifa, T.; Ayoubi, S.; Shaban, K.B. Surviving link failures in multicast VN embedded applications. In Proceedings of the IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, 25–29 April 2016; pp. 645–651.
17. Qiang, Z.; Qiang, W.; Sheng, F.; Wu, L. Heuristic survivable virtual network embedding based on node migration and link remapping. In Proceedings of the IEEE 7th Joint International Information Technology and Artificial Intelligence Conference, Chongqing, China, 20–21 December 2014; pp. 181–185.
18. Shahriar, N.; Ahmed, R.; Khan, A.; Chowdhury, S.R.; Boutaba, R.; Mitra, J.; David, R. ReNoVatE: Recovery from node failure in virtual network embedding. In Proceedings of the 12th International Conference on Network and Service Management (CNSM), Montreal, QC, Canada, 31 October–4 November 2016; pp. 1–9.
19. Shahriar, N.; Ahmed, R.; Chowdhury, S.R.; Khan, A.; Boutaba, R.; Mitra, J. Generalized recovery from node failure in virtual network embedding. *IEEE Trans. Netw. Serv. Manag.* **2017**, *14*, 261–274. [CrossRef]
20. Gong, S.; Chen, J.; Zhao, S.; Zhu, Q. Virtual network embedding with multi-attribute node ranking based on TOPSIS. *KSII Trans. Int. Inf. Syst.* **2016**, *10*, 522–541. [CrossRef]
21. Su, Y.; Meng, X.; Meng, Q.; Zhao, Z. Environment adaptive and joint topology aware virtual network embedding algorithm. *J. Electr. Inf. Technol.* **2018**, *40*, 79–86. [CrossRef]
22. Zhu, Y.; Ammar, M. Algorithms for assigning substrate network resources to virtual network components. In Proceedings of the 25th IEEE International Conference on Computer Communications, Barcelona, Spain, 23–29 April 2006; pp. 1–12.

23. Gong, S.; Chen, J.; Huang, C.; Zhu, Q. Trust-aware secure virtual network embedding algorithm. *J. Commun.* **2015**, *36*, 180–189. [CrossRef]
24. Zhao, Z.; Meng, X.; Su, Y.; Li, Z. Virtual network embedding based on node connectivity awareness and path integration evaluation. *KSII Trans. Int. Inf. Syst.* **2017**, *11*, 3393–3412. [CrossRef]