IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 4, 2022, accepted June 1, 2022, date of publication June 8, 2022, date of current version June 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3180744

Fault-Tolerant Embedding Algorithm for Node Failure in Airborne Tactical Network Virtualization

JINGCHENG MIAO, NA LV, QI GAO, KEFAN CHEN, AND XIANG WANG
School of Information and Navigation, Air Force Engineering University, Xi'an 710077, China

Corresponding author: Na Lv (lvnn2007@163.com)

This work was supported by the Shaanxi Provincial Natural Science Foundation under Grant 2020JQ-483.

ABSTRACT Airborne tactical networks (ATNs) are driving the promising development of Internet of battle Things (IoBT) by enabling efficient information sharing, which is impeded by the network ossification problem due to the tightly coupled network architecture. As a solution, network virtualization (NV) can solve the ossification problem by breaking the tight coupling between applications and network infrastructure for ATNs. With complex interference and malicious attacks, the application of NV is challenged by network failures when instantiating virtual networks on a shared substrate network, which is known as survivable virtual network embedding (SVNE). However, existing SVNE algorithms, mostly designed for wired networks, are not necessarily optimal for the virtualization of ATNs due to the complex wireless interference. To this end, a fault-tolerant SVNE algorithm, termed SVNE-FT, is proposed to recover virtual networks from single node failure (end or switching node failure) under the complex wireless interference. To end node failure, SVNE-FT adopts a novel node ranking approach to select reliable substrate nodes for virtual nodes and remaps the failed virtual nodes by releasing part of the substrate paths to improve the resource utilization. In addition, to switching node failure, it adopts the improved pre-configured cycle (p-Cycle) technology to augment the reliable link mapping with differentiated p-Cycles that protect switching node and reduce the resource consumption of backups. Numerical simulation results reveal that SVNE-FT outperforms typical and latest heuristic SVNE algorithms under the complex interference of ATNs. For instance, average acceptance ratio of virtual networks improves at least 12%.

INDEX TERMS Airborne tactical network, network virtualization, survivable virtual network embedding, node failure, pre-configured cycle.

I. INTRODUCTION

The revolution of Internet of battle Things (IoBT) [1] and space-air-ground integrated networks [2] prompts the reshaping of military communication systems, where airborne tactical networks (ATNs) exploit manned or unmanned aircrafts with various tactical capabilities [3]. Therefore, ATNs have recently gained significant attention in the push to rapidly deploy on-demand air resources to maintain coverage and provide reach-back to military aircrafts as well as surface and space military units [4]. However, in airborne field, sharing tactical information reliably is challenging due to the network node failure caused by complex interference and malicious attacks. Moreover, current ATNs are application coupled with vertical integration of customized software stack and hardware, causing insufficient interoperability and high upgrading difficulty [5], [6]. This is known as network ossification problem, impeding the efficient resource allocation to recover from node failure and the revolution of information communication technology in ATNs. Therefore, abstracting network resources is compelling to support multi-mission in developing ATNs against node failure.

Network virtualization (NV), solving the network ossification problem, is evolving as a key technology for emerging areas like cloud computing [7] and smart IoT [8]. In NV, multiple heterogeneous virtual networks (VNs) can coexist and seamlessly share underlying networking resources on the same substrate network (SN) [9]. Thus, NV can provide a high level of resource sharing among various mission applications of ATNs. To implement NV, software defined

The associate editor coordinating the review of this manuscript and approving it for publication was Amjad Ali.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

networking (SDN) can provide an ideal platform in comparison with conventional methods since it offers global state view of the resource usage and strict quality of service (QoS) provisioning without tunnelling overhead [10]. Therefore, a viable platform can be provisioned for the virtualization of ATNs, called as airborne tactical network virtualization (ATNV).

To realize the benefits of ATNV, resource management challenges exist in mapping virtual nodes (VNodes) and links (VLinks) of VNs onto substrate nodes (SNodes) and links (SLinks), respectively. This is known as VN embedding (VNE) problem that is proved to be NP-hard [11]. VNE solutions are also requested to survive substrate failures to avoid degraded QoS, which is widely accepted as survivable VNE (SVNE) [12]. In NV, the survival of SNode failure is fairly challenging due to the shared nature of SN. To single node failure, there are two kinds of node failure, namely, end and switching node failure. End node failure is the failure of SNode that maps VNode, whereas, switching node failure is the failure of a switching SNode on the substrate path that maps VLinks. Thus, single node failure can cause the malfunction of multiple VNodes and VLinks, seriously degrading the service performance of VNs. SVNE has received significant attention from the research community [13]–[17]. The existing SVNE solutions are broadly classified into two categories: proactive [13], [14] and reactive mechanisms [15]–[17]. The proactive mechanisms, with high resource expense, provision disjoint redundant resources as backups, offering fast recovery from failures. In addition, the reactive mechanisms, with lower resource expense, remap the failed VNodes and VLinks as failures happen. However, to our knowledge, there are no wireless SVNE algorithms that focus on single node failure. These wired SVNE algorithms against node failure do not account for wireless network dynamics (e.g., link quality instability). In addition, they do not achieve a performance balance between backup resources and recovery efficiency.

Based on these backgrounds, we focus on the wireless SVNE problem in the virtualization of ATNs and propose a fault-tolerant SVNE algorithm, termed SVNE-FT, that considers both reactive and proactive mechanisms to recover VNs from single node failure under the wireless environment. To end node failure, we propose a novel node ranking method to select reliable SNodes to map VNodes and adopt a reactive recovery strategy to efficiently remap failed VNodes. In addition, to switching node failure, we reliably construct differentiated p-Cycles to protect the vulnerable switching nodes of substrate paths that map VLinks. To further validate the strength of SVNE-FT, we conduct a comprehensive simulation. Numerical results reveal that SVNE-FT outperforms representative and latest SVNE algorithms in terms of acceptance ratio and revenue to cost ratio. Also, it greatly enhances the achievable recovery efficiency, decreasing recovery delay.

Main contributions of this paper are summarized as follows:

1) A heuristic SVNE scheme is proposed to recover from end or switching node failure for the virtualization of ATNs. To end node failure, SVNE-FT adopts a novel reactive approach to efficiently remap the affected virtual nodes. Then, to switching node failure, it constructs reliable and differentiated p-Cycles for unreliable switching SNodes of the affected VLinks by the improved p-Cycle technology.

2) To reliably map VNodes, node capacity, link bandwidth, wireless interference, recent failure time and node failure times are exploited when ranking SNodes. While in previous wireless mapping scheme [18], [19], virtual node constraints only include node capacity and link bandwidth.

3) Comprehensive simulations are made to validate the advantage of SVNE-FT. We compare our heuristic algorithm not only with its reactive version without p-Cycle technology, but also with typical SVNE algorithms. In addition, simulation results are illustrated in order to highlight our proposed algorithm.

The rest of the paper is organized as follows. Section II is the related work. Then, Section III presents the SDN based model for ATNV and formulates the SVNE problem. Next, Section IV presents the details of SVNE-FT. The simulation work is implemented in Section V, along with parameters setting. Finally, Section VI concludes our findings.

II. RELATED WORK

ATNs are currently constructed on discrete data link systems (e.g., Link-16, TTNT, MADL) [6]. These communication systems have been a critical information sharing capability between both manned and unmanned military aircraft as well as surface and ground platforms for a number of decades. While these systems serve the immediate need, they fail to fulfil the actual desire to support an ever-increasing number of users and emerging applications, and then adapt to the network failures [3]. This is because of the ossification problem in the vertically integrated network architecture, which also exists in many public networks like unmanned aerial vehicles (UAVs) networks. The virtualization technology has been a promising solution in UAVs [20], [21]. Therefore, NV can improve the resource allocation in recovering node failure during missions for ATNs by solving the SVNE problem.

The SVNE problem, proved to be NP-hard, has been accepted as one key research area to improve the survivability of VNs affected by network failures. Therefore, many efforts have been made in heuristic approaches. Rahman *et al.* in [12] firstly address the survivability of VNs. There are many significant aspects in the research literature such as single SNode failure, multiple SNode failures, single SLink failure and multiple SLink failures [22]. This paper focuses on the single SNode failure. After a VN is embedded into the SN, single node failure can have direct and bad influence in the normal operation of virtual nodes and links. Therefore, the approaches for ensuring survivability of VNs during a single substrate node failure can be broadly classified into two categories, namely, end node and switching node protection approaches. In the following subsections, we discuss the

end node (Section II-A) and switching node (Section II-B) protection approaches from the SVNE literature.

A. PREVIOUS TYPICAL AND LATEST END NODE PROTECTION APPROACHES

To one embedded VN, VNodes are mapped onto qualified SNodes, called as end nodes for transmitting and receiving the data on the virtual links. End node protection approaches protect end nodes from single node failure. On the one hand, many proactive approaches (e.g., [13], [23]) have been proposed. Liu *et al.* address the problem of selecting standby virtual routers that is provided by the SN to serve VNs under a single node failure. A node protection approach is proposed in this paper to dynamically reconfigure VNs back to original topologies right after a single substrate node failure. In addtion, for the multiple substrate node failures, Xiao *et al.* [23] use topology attributes, considering dedicated substrate backup scheme, to enhance the survivable VN. In these proactive protection approaches, resource utilization may be reduced due to the fact that backup resources remain unused during normal operations. On the other hand, reactive approaches (e.g., [17], [24]) are also proposed to protect end nodes. Shahriar *et al.* [17] propose a reactive protection scheme, solving the problem of recovering a batch of VNs affected by a substrate node failure. This scheme aims to achieve customized objectives such as fair treatment on the failed VNs, partial treatment based on priority, and so on. In contrast, Chang *et al.* [24] propose a migration aware protection approach to recover from a single substrate node failure. The approach facilitates the re-mapping of the failed virtual nodes and virtual links by the migration of some active virtual nodes and links to free up some substrate resources. All of these reactive approaches may need a chain of migrations to converge and thus disrupting ongoing communication in the VN.

Most of these end node protection approaches are designed for wired networks. For the specific wireless environment in ATNs, it is definitely requested to reliably map virtual nodes, considering the complex electromagnetic interference. In addition, proactive approaches will cause high resource expense by prepare backups for both virtual nodes and their adjacent virtual links, which may be not proper for ATNs with limited resources. Thus, to improve the resource utilization, it is important to design reactive approaches to protect virtual node from single node failure.

B. PREVIOUS TYPICAL AND LATEST SWITCHING NODE PROTECTION APPROACHES

To one embedded VN, VLinks are mapped onto the corresponding substrate paths including switching nodes that forward the data of virtual links. A single substrate node failure on switching nodes may cause the operation disruption of multiple virtual links. However, switching nodes are not fully taken into consideration in the node protection for the survivability of VNs. Most of the previous SVNE algorithms deal with the switching node failure by remapping the virtual links affected by node failure, which will increase the recovery delay and fail to find the proper path if most of the network resources of SN have been used. Thus, there are proactive approaches [25], [26] that protect VLinks from switching node failure, adopt p-Cycle technology. Jarray *et al.* [25] address the problem of single/multiple logical link failures caused by a single substrate node failure in switching substrate node. A two-step protection mechanism is proposed to recover VNs. The first step uses a large-scale optimization technique to embed VNs. And the second step adopts link p-Cycle based protection techniques to protect the switching substrate nodes while minimizing the backup resources. Also, the logical layer related to the VN topology is considered to guarantee a node failure independent path protection scheme. In [26], Jarray *et al*. extend the results to handle any single physical failure in the substrate by adopting p-cycle protection approach in physical layer. These switching node protection approaches are mostly designed for wired networks and thus cannot adapt well to ATNs due to the wireless environment. In addition, p-Cycle technology also takes too much network resources to protect the switching node.

Most of these switching node protection approaches are based on reactive schemes, which cannot efficiently protect switching node protection. However, proactive schemes, especially for p-Cycle based approaches, can offer high efficiency in recovering VNs from switching node failure for the virtualization of ATNs. In addition, it is vital to fully consider the complex interference while protecting switching nodes and the balance between backup resources and recovery efficiency due to the limited network resources.

C. BRIEF SUMMARY OF PREVIOUS WORKS

To summarize, the ossification problem of ATNs can be properly solved by NV. Also, SVNE problem has been accepted as a vital aspect of NV to improve the network reliability. Thus, it is significant to further study SVNE problem that is NP-hard for the virtualization of ATNs to protect mission applications from network failures. In virtualization environment, a single SNode failure can result in the operation disruption of both VNodes and multiple VLinks. Many efforts in the literature have been made to protect end nodes (Section II-A) and switching nodes (Section II-B) against a single node failure. However, these heuristic node protection algorithms are designed for wired networks and may not adapt to the specific wireless environment in ATNs. In addition, these node protection algorithms do not consider the balance between backup resources and recovery efficiency which is important for the virtualization of ATNs with the limited network resources.

Our proposed SVNE-FT algorithm differs from previous heuristic SVNE algorithms in four main aspects. First of all, the SVNE-FT algorithm adopts a reliable embedding method to map VNs, decreasing the effects of a single substrate node failure on the VN layer. Second, for end node protection, we design a proactive mechanism that shortens

the chain of migrations to converge. Next, a p-Cycle based method is adopted in switching node protection, considering the specific complex interference in air-combat field. Finally, a comprehensive simulation is implemented against typical node-protection heuristic SVNE algorithms under wireless interference.

III. PRELIMINARIES

In this section, we firstly present the slice-based ATNV model. Then, we formally represent the SVNE problem model. Finally, evaluation metrics and objective functions are defined.

A. SLICE-BASED ATNV MODEL

ATNs refer to large-scale wireless multi-hop networks, formed by manned and unmanned military aircrafts as well as surface and ground platforms. Existing ATNs can support a certain degree of operational coordination between aviation platforms in the existing aviation combat mode in terms of network transmission performance indicators such as transmission reliability, end-to-end delay, and transmission rate [27]. However, the ossification problem is impeding the revolution of ATNs to support the application requirements of future aviation swarm combat. The ossification problem lies in the network architecture that is tightly coupled and vertically implemented to support users and applications. Therefore, it is vital to design a scalable and flexible infrastructure. A slice-based ATNV framework is designed in Fig. 1, inspired by the SDN-enabled ATN [27] and the virtualization scheme [18]. The framework is composed of the following layers:

1) Substrate network layer: Massive aircrafts are deployed in concomitant formation. Lead aircrafts in the formation form a wireless multi-hop network, abstracted as the shared airborne tactical substrate network (ATSN). These aircrafts, equipped with SDN-enabled devices, are the access points that access and transfer situation information and thus compose the data plane.

2) Virtualization layer: A hypervisor virtualizes the ATSN and performs as a slicing agent between data plane and SDN controllers. The cloud icon represents the tactical data centre that owns strong information processing capability to deal with the rapid changes of situation information and mission applications. The hypervisor can orchestrate and collaborate multiple applications in an overlapped way and thus adopts the SVNE algorithm. Also, the hypervisor can adopt the optimized link state routing (OLSR) protocol [28] to collect the dynamic link state information and support the resource requests of mission applications stably and timely.

3) Application layer: Various military mission applications are precisely abstracted into logical and self-contained airborne tactical virtual networks (ATVNs). Accordingly, each ATVN has arbitrary topology with customized QoS demands to fulfil specific tactical service demands. Also, a dedicated SDN controller is deployed to support the protocol reconfiguration and evolution of each ATVN.

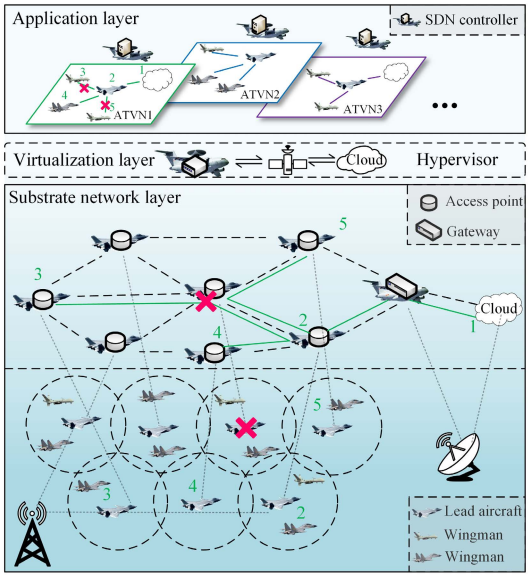


FIGURE 1. Illustration of slice-based virtualization for ATN.

In general, there are four main features in the virtualized ATNs. First of all, a platform is offered to accept and deploy military applications since each ATVN is allowed to run and evolve its own protocol. Therefore, network services are flexibly coupled with military airborne missions, solving the ossification problem of traditional ATNs [29]. Second, the hypervisor possesses global state view of the resource usage to make the global allocation possible. Third, the full isolation exists in the network configurations of different ATVNs [30], satisfying the efficiency and security demand of military missions. Finally, its high scalability and sustainability can decrease the overhead of extra devices for innovative mission applications. Thus, new network technologies can be deployed more quickly and easily [9].

B. SVNE PROBLEM MODEL

1) AIRBORNE TACTICAL SUBSTRATE NETWORK (ATSN)

We represent ATSN as an undirected graph $G^S = (N^S, L^S)$, where N^S and L^S denote the set of SNodes and SLinks, respectively. Each SNode $n \in N^S$ is served as an access point for lead aircraft associated with radio transmission power p^S and location $loc(n)$. p^S consists of basic power $p_b(n)$, used within aviation formation, and cooperation power $p(n)$, used between aviation formations. $loc(n)$ is assumed to be relatively unchanged during the embedding process. To each SLink $l^S \in L^S$, $b(l^S)$ and $loc(l^S)$ are bandwidth capacity and relative location, respectively. l_{ij}^S is a SLink that directly connects n_i and neighbouring n_j. p_{ij}^S is any loop-free substrate path between n_i and n_j. N_f^S and L_f^S are the set of failed SNodes and SLinks, respectively. The down part of Fig. 2 presents a general ATSN. The numbers next to SNodes are $p(n)$. Also, the numbers over SLinks are $b(l^S)$.

2) AIRBORNE TACTICAL VIRTUAL NETWORK (ATVN)

ATVN is represented as an undirected graph $G^V = (N^V, L^V)$, where N^V and L^V are the sets of VNodes and

VLinks, respectively. To VNode $v \in N^V$, $loc(v)$ is the location of corresponding aircraft (lead aircraft or wingman) where the remote command and control platform collects or transmits situation information. φ^V is the coverage radius of v. To VLink $l^V \in L^V$, it has the limited transmission rate $TR\left(l^V\right)$ since military missions require that tactical information must be transmitted as fast as possible, which is closely related to the transmission rate that a SLink can offer. l^V_{mn} is a VLink that connects v_m and its neighboring v_n. N^V_f and L^V_f are the set of failed VNodes and VLinks, respectively. The up part of Fig. 2 shows two ATVN requests with specific topologies. The numbers over VLinks are $TR\left(l^V\right)$.

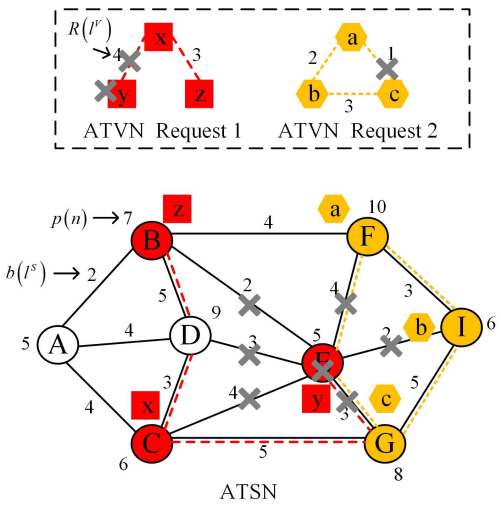


FIGURE 2. ATVN embedding and impacts of failure.

3) EMBEDDING PROCESS

Embedding any ATVN onto ATSN includes node and link mapping. To node mapping, v from the same ATVN is mapped onto a different n, which is expressed by M^N: $v \in N^V \rightarrow n \in N^S$. In Fig. 2, node mapping results of ATVN request 1 and 2 are $M^N(x) = C$, $M^N(y) = E$, $M^N(z) = B$, $M^N(a) = F$, $M^N(b) = I$ and $M^N(c) = G$. Next, to link mapping, $l^V \in L^V$ from the same ATVN is mapped onto a substrate path, which is expressed by M^L:$l^V_{sd} \rightarrow p^S_{ij}$, where $\left(n_i, n_j\right)$ are SNodes assigned to (v_s, v_d) source and destination VNodes of l^V_{sd}, respectively. As shown in Fig. 2, link mapping results are $M^L((x,y)) = \{(C,G),(G,E)\}$, $M^L((x,z)) = \{(C,D),(D,B)\}$, $M^L((a,b)) = \{(F,I)\}$, $M^L((a,c)) = \{(F,E),(E,G)\}$ and $M^L((b,c)) = \{(I,G)\}$.

4) TYPES OF FAILURE

The failure of a SNode $n \in N^S_f$ can directly result in a set of VNode and VLink failures, defined as $N^V_f = \left\{v \in N^V \middle| n = M^N(v) \in N^S_f\right\}$ and $L^V_f = \left\{l^V \in L^V \middle| l^S \in M^L\left(l^V\right) \wedge l^S \in L^S_f\right\}$. There are two types of direct failures of ATVNs:

VNode failure: The VNode $v_m \in N^V_f$, mapped onto the failed $n \in N^S_f$, will directly malfunction as end node failure happens, which also accordingly cause the indirect failures of adjacent VLinks, represented by $vl^f_m = \left\{l^V_{mn} \middle| v_m \in N^V_f\right\}$. In Fig. 2, as SNode E fails, VNode y malfunctions directly and VLink (x, y) fails indirectly.

Independent VLink Failure: The VLink l^V_{sd}, mapped onto a substrate path $M^L\left(l^V_{sd}\right)$ including the failed SNode $n \in N^S_f$, will directly fails as switching node n fails. It is represented by $E^f = \left\{l^V_{sd} \middle| n \in M^L\left(l^V_{sd}\right) \wedge n \in N^S_f \wedge v_s \notin N^V_f \wedge v_d \notin N^V_f\right\}$. In Fig. 2, VLink (a, c) fails as switching SNode E fails.

C. EVALUATION METRICS

The average acceptance ratio is an important metric that presents the efficiency of successfully accepting VN requests. It is shown in Eq. (1) below.

$$\eta_{suc} = \lim_{T\rightarrow\infty} \frac{\sum_{t=0}^{T} NUM_{Emb}(t)}{\sum_{t=0}^{T} NUM_v(t) + \delta} \tag{1}$$

where $NUM_{Emb}(t)$ is the number of accepted ATVN requests at time t. $NUM_v(t)$ is the total number of ATVNs. δ represents a constant infinitely close to 0.

The longterm average revenue to cost ratio can present the ability in resource utilization, especially for link mapping. It is shown in Eq. (2) below.

$$R/C = \lim_{T\rightarrow\infty} \frac{\sum_{t=0}^{T} \sum_{G^V \in VN_{map}(t)} R\left(G^V, t\right)}{\sum_{t=0}^{T} \sum_{G^V \in VN_{map}(t)} C\left(G^V, t\right)} \tag{2}$$

where $VN_{map}(t)$ is the working ATVNs at time t. The revenue $R\left(G^V, t\right)$ represents the total network resource requirement (node power and link bandwidth resource) of embedding accepted ATVNs at time t. Under the total network resource requirement, it is useful to calculate the amount of consumed network resource that is represented by the cost $C\left(G^V, t\right)$. Eq. (3) and (4) are the specifical definition of $R\left(G^V, t\right)$ and $C\left(G^V, t\right)$, respectively.

$$C\left(G^V, t\right) = \alpha \sum_{v\in N^V} p(n)|_{v\rightarrow n} + \sum_{l^V\in L^V} \sum_{l^S\in p^S} b\left(l^S\right)|_{l^V\rightarrow p^S} \tag{3}$$

$$R\left(G^V, t\right) = \beta \sum_{v\in N^V} p(n)|_{v\rightarrow n} + \sum_{l^V\in L^V} \sum_{l^S\in p^S} \frac{b\left(l^S\right)|_{l^V\rightarrow p^S}}{\left|p^S\right|} \tag{4}$$

where α and β are weighting coefficients that coordinate the relationship between node power and link bandwidth. In this paper, both α and β are set to be 1. $p(n)|_{v\rightarrow n}$ is the power resource allocated by n to map v. With VLink l^V mapped onto path p^S, $b\left(l^S\right)|_{l^V\rightarrow p^S}$ is the bandwidth resource allocated by a SLink l^S of p^S to map l^V. $\left|p^S\right|$ is the total link number of p^S.

The average failure recovery ratio presents the ability in recovering the failed SNodes. It is shown in Eq. (5).

$$a\_r = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} num_{f\_r}(t)}{\sum_{t=0}^{T} num_f(t) + \bar{\delta}} \tag{5}$$

where $num_f(t)$ represents the total number of failures at time t. $num_{f\_r}(t)$ denotes the number of failures that are successfully recovered at time t. Similar to δ in Eq. (1), $\bar{\delta}$ represents a constant infinitely close to 0.

D. OBJECTIVE FUNCTIONS

The major objective of the proposed algorithm is to maximize the average acceptance ratio under single node failure. The objective function is formulated as given below.

Objection

$$Maximize \left\{ \lim_{T\to\infty} \frac{\sum_{t=0}^{T} NUM_{Emb}(t)}{\sum_{t=0}^{T} NUM_{Arr}(t)} \right\} \tag{6}$$

Constraints

$$\forall v \in N^V, \quad \forall n \in N^S, \; x(v,n) = \begin{cases} 1, & iff \; n = M^N(v) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

$$\sum_{n\in N^S} x(v,n) \le 1 \tag{8}$$

$$x(v,n) \times dist(v,n) \le \varphi^V \tag{9}$$

$$VNoC(v) \times x(v,n) \le SNoC(n) \tag{10}$$

$$\forall l^V \in L^V, \quad \forall l^S \in L^S, \; x\left(l^V, l^S\right) = \begin{cases} 1, & iff \; l^S \in M^L\left(l^V\right) \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

$$\sum_{l_t^V \in L_t^V} VLiC\left(l^V\right) \times x\left(l^V, l^S\right) \le SLiC\left(l^S\right) \tag{12}$$

$$\sum_{l_{ij}^S \in L^S} x\left(l_{um}^V, l_{ij}^S\right) - \sum_{l_{ji}^S \in L^S} x\left(l_{um}^V, l_{ji}^S\right) = \begin{cases} 1, & x(v_u, n_i) = 1 \\ -1, & x(v_m, n_i) = 1 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Constraints work when mapping VNodes and VLinks of ATVNs. First, the constraints of node mapping are included in Eq. (7) to (10). In Eq. (7), $x(v,n)$ is the Boolean variable that represents if VNode v is mapped onto SNode n. Eq. (8) ensures that n can only map at most one v of the same ATVN. Eq. (9) ensures that the distance $dist(v,n)$ between v and n must be within the radius φ^V. In Eq. (10), $VNoC(v)$ represents the practical node capacity requirement of v if $x(v,n) = 1$ and $SNoC(n)$ represents the existing node capacity of n that excludes the node power resource used by constructing p-Cycles. Eq. (10) ensures that n must have more capacity than the requirement of v. Next, the constraints of link mapping are included in Eq. (11) to (13). In Eq. (11), $x\left(l^V, l^S\right)$ is also the Boolean variable that represents if l^S is one SLink of the path $M^L\left(l^V\right)$. In Eq. (12), $VLiC\left(l^V\right)$ represents the practical link capacity requirement of l^V that includes the link bandwidth resource used by both mapping VLinks and constructing p-Cycles. $SLiC\left(l^S\right)$ represents the existing link capacity of l^S. Eq. (12) ensures that l^S must meet the link capacity of all the VLinks that are mapped onto it. Eq. (13) is the connectivity constraint that refers to the flow conservation constraint for routing one unit of net flow from v_u to v_m.

IV. HEURISTIC SOLUTION: SVNE-FT

Due to the intractability of SVNE problem, we resort to a heuristic algorithm, SVNE-FT (Fig. 3), that is detailed in this section. Firstly, after one ATVN request arrives, SVNE-FT will adopt **Algorithm 1** to map VNodes and VLinks. If the embedding process is successful, the node failure detection will start to check if there is a single node failure. Then, if single node failure happens, SVNE-FT will adopt **Algorithm 2** to deal with the affected VNodes and VLinks. Next, in **Algorithm 1**, node and link mapping are coordinated in one-stage. Meantime, it reliably maps VNodes (**Procedure 1**) and VLinks (**Procedure 3**) against end and switching node failure, respectively. Finally, in **Algorithm 2**, the recovery strategy is adopted to deal with end node (**Procedure 2**) or switching node (**Procedure 4**) failure.

To clarify the efficiency of the improved p-Cycle protection technology (**Procedure 3**), it is wise to make a simple reactive version of SVNE-FT, SVNE-FTs, that deletes **Procedure 3** and **Procedure 4** and only remaps failed switching SNodes and its adjacent SLinks.

A. PROTECTION AGAINST END NODE FAILURE

For an ATVN request $G^V = \left(N^V, L^V\right)$, $M^N(v) : v \to n$ denotes the process of mapping VNode v onto ATSN. This subsection introduces a novel reliable node ranking approach and a recovery strategy against end node failure.

1) RELIABLE NODE RANKING

To select the proper VNode, VNodes are ranked into a list based on ranking value that indicates the capacity demand. In this VNode list (VNo_List), the ranking value is expressed as

$$NoV(v) = \sum_{l^V \in L_v^V} TR\left(l^V\right) \tag{14}$$

where L_v^V denotes the set of VLinks adjacent to v. $NoV(v)$ is the total requirement of transmission rate $TR\left(l^V\right)$ for all adjacent VLinks of v. The VNode with the highest ranking value will be selected and mapped.

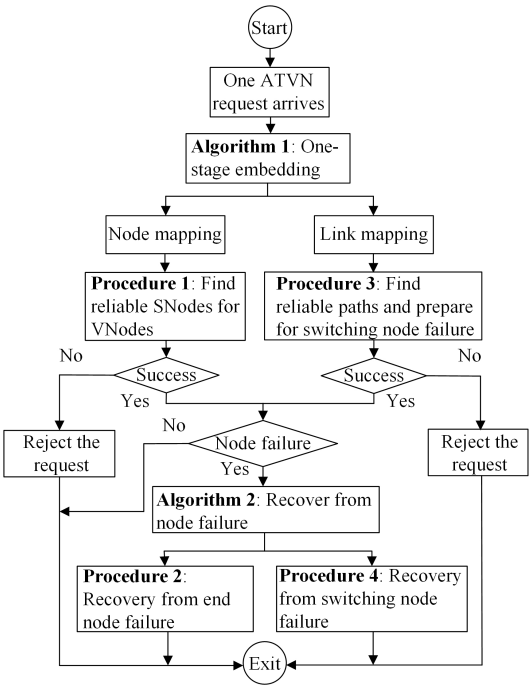


FIGURE 3. Flow chart of SVNE-FT.

Procedure 1 Reliable Node Ranking

Input: $G^V = \left(N^V, L^V\right)$, $G^S = \left(N^S, L^S\right)$, a set of selected VNodes;
Output: VNode v with corresponding SNo_List;
1. **for** any v in the set of selected VNodes **do**
2. **for** all the VLinks in L_v^V **do**
3. Calculate $NoV\ (v)$ with Eq. (14);
4. Record v with $NoV\ (v)$ in VNo_List;
5. **end for**
6. **end for**
7. Select v with the highest $NoV\ (v)$ in VNo_List;
8. **for** SNode n meeting Eq. (9) for v **do**
9. **for** SLink l^S in L_n^S **do**
10. Calculate $TR(l^S)$ with Eq. (17);
11. **end for**
12. Ignore SNodes that cannot meet Eq. (10);
13. Calculate $NoV(n)$ with Eq. (15);
14. Record n with $NoV\ (n)$ in SNo_List;
15. **end for**
16. **if** No SNodes meet Eq. (9) and (10) **do**
17. Reject the ATVN request.
18. **end if**

To select the proper SNode, a SNode list (SNo_List) is formed based on SNode ranking value that is expressed as

$$NoV\ (n) = SF\ (n) \sum_{l^S \in L_n^S} TR(l^S) \tag{15}$$

where L_n^S is the set of SLinks adjacent to SNode n. $SF\ (n)$ denotes the safety factor against single node failure, which is expressed as

$$SF\ (n) = \begin{cases} \dfrac{pos\,[n]}{\left|N^S\right| \cdot k}, & k > 0 \\ 1, & k = 0 \end{cases} \tag{16}$$

where k denotes the number of times that SNode n fails over a period of time. If $k = 0$, n owns high reliability and then $SF\ (n)$ is set to be 1. $pos\,[n]$ denotes the order number of n in a list of failed SNodes in chronological order, which means the distance to the first failed SNode [31]. $\left|N^S\right|$ represents the total number of SNodes in ATSN. Thus, if n works normally or has less frequency of failure, it will have a higher $SF\ (n)$.

In Eq. (15), $TR(l^S)$ denotes the highest transmission rate that SLink l^S can offer. According to the famous *Shannon formula*, we can quantify the interactions between interference and network resource to get a specific value of transmission rate for one SLink in the unstable air-combat field. To simplify the theoretical analysis, we only deal with attenuation and large-scale fading in this paper. Thus, $TR(l^S)$ can be expressed as [5]

$$TR(l^S) = b(l^S) \log_2 \left(1 + \frac{p\ (n)\ g\left(l^S\right)}{AveInf\ (n) + AveInf\ \left(l^S\right)}\right) \tag{17}$$

where $b(l^S)$ and $p\ (n)$ denote available bandwidth resource of l^S and power resource of n, respectively. $g\left(l^S\right)$ denotes the link gain. Also, $AveInf\ (n)$ and $AveInf\ \left(l^S\right)$ denote average node and link interference, respectively.

The specific procedures are shown in **Procedure 1**. As for the time complexity, let $|N^S|$ denote the total number of SNodes. The inner loop (step 6 to 8) of requires at most $O(|N^S|^2)$ since there are at most $|N^S| - 1$ number of SLinks adjacent to one SNode. Therefore, in the worst case scenario, the time complexity of **Procedure 1** is $O(|N^S|^2)$.

2) RECOVERY FROM END NODE FAILURE

In Fig. 4, VLink l_{um}^V is originally mapped onto substrate path $\{(A, D)\ , (D, E)\ , (E, I)\}$ with VNode u and m mapped onto SNode A and I, respectively. The failure of end node A causes the indirect malfunction of its adjacent SLinks (A, B), (A, D) and (A, C). In the virtual layer, u and l_{um}^V accordingly fail.

It is essential to recover u and l_{um}^V as fast as possible to complete military missions. In the reactive recovery approach, u can be remapped onto SNode C. Then, l_{um}^V can be partly remapped onto substrate path $\{(C, G)\ , (G, E)\ , (E, I)\}$ because (D, E) and (E, I) are released and (D, E) is occupied by other VLinks. To meet the resource requirement of l_{um}^V, it will take some time to allocate the link bandwidth and node power resource on the new path. In some cases, the recovery will fail if the remaining resource on $\{(C, G)\ , (G, E)\}$ is not enough. To decrease the time spending, we assume that original SLinks are not released. Then, SLink (C, D) can constitute the path $\{(C, D)\ , (D, E)\ , (E, I)\}$ that remaps l_{um}^V. The time spent in resource allocation of SLink (C, D) can be shorter.

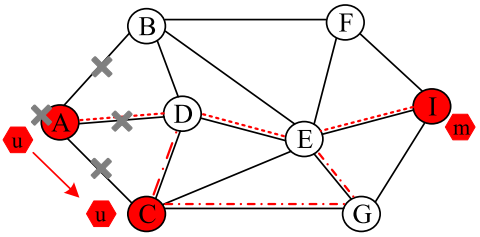


FIGURE 4. Remap VNode against end node failure.

Procedure 2 Recovery From End Node Failure

Input: $G^V = (N^V, L^V)$, a failed VNode v, the adjusted $G^S = (N^S, L^S)$ that excludes the failed SNode and SLinks;
Output: The mapping results of v with corresponding VLinks;
1. Run ***Procedure 1*** for v;
2. **if** the ATVN request is rejected **do**
3. The recovery fails and **break**;
4. **end if**
5. Get the SNo_List of v;
6. **for** each SNode n in the SNo_List **do**
7. Get $NoV_r(n)$ based on Eq. (18) and (19);
8. **end for**
9. Select the SNode n with highest $NoV_r(n)$ and remap v onto n;
10. **for** each l^V in the VLink set L_v^V **do**
11. Hold the normal SLinks of p^S;
12. Find the shortest reliable path that connects n and p^S;
13. **if** the path cannot be found **do**
14. The recovery fails and **break**;
15. **end if**
16. **end for**

VNode b in Fig. 2, not like u in Fig. 4, connects to more than one VLink. In this case, to decrease the time spending, the remapping of b needs to consider the distance between the selected SNode and extant SLinks of previous substrate paths. The distance can be expressed as

$$\forall v \in N_f^V, \quad n \in \text{SNo\_List}$$
$$Dr(n) = \sum_{l^V \in L_v^V} dr\left(n, p^S \middle| l^V \to p^S\right) \tag{18}$$

where $dr\left(n, p^S\right)$ represents the shortest hoops between SNode n and substrate path p^S. The node ranking value can be modulated into

$$NoV_r(n) = \frac{SF(n)}{Dr(n)} \sum_{l^S \in L_n^S} TR(l^S) \tag{19}$$

The specific procedures are shown in **Procedure 2**. The time complexity, mainly determined by the loop (step 10 to 14), is at most $O(|L^V|(|L^S| + |N^S|\log(|N^S|)))$.

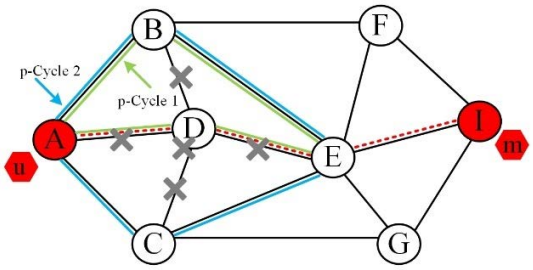


FIGURE 5. Protection against switching node failure.

B. PROTECTION AGAINST SWITCHING NODE FAILURE

For $G^V = (N^V, L^V)$, the mapping of one VLink l_{um}^V is expressed as $M: l_{um}^V \to p_{ij}^S$. VNodes v_u and v_m can be mapped onto SNodes n_i and n_j by adopting **Procedure 1**, respectively. Based on our gained knowledge and previous research results, the reliable link mapping approach [32] is adopted in consideration of complex interference. This subsection presents the p-Cycles based recovery strategy against switching node failure.

1) SWTICHING NODE PROTECTION APPROACH

In Fig. 5, a switching node failure hits SNode D, indirectly causing failures of adjacent SLinks. D can be protected by p-Cycle 1 or p-Cycle 2. In p-Cycle 1, D is an on-cycle node, protected by pre-configured path $\{(A,B),(B,E)\}$. As D fails, l_{um}^V can be remapped onto $\{(A,B),(B,E),(E,I)\}$. In p-Cycle 2, D is a straddling node, protected by both $\{(A,B),(B,E)\}$ and $\{(A,C),(C,E)\}$. As D fails, l_{um}^V can be remapped onto $\{(A,B),(B,E),(E,I)\}$ or $\{(A,C),(C,E),(E,I)\}$.

With limited network resource, it is not proper to construct p-Cycles, occupying power and bandwidth resources, for all switching SNodes for VLinks. Thus, we set a decision variable as

$$NoV\%(n) = \frac{\forall n_i \in N^S \quad NoV(n) - \min\{NoV(n_i)\}}{\max\{NoV(n_i)\} - \min\{NoV(n_i)\}} \tag{20}$$

where $NoV\%(n)$ is the normalized $NoV(n)$. With $NoV\%(n)$ less than Γ_1 and more than Γ_2, n can be protected by p-Cycle 1. In addition, it can be protected by p-Cycle 2 if $NoV\%(n)$ is less than Γ_2 or n has once failed.

To construct a p-Cycle for n, we first find its two adjacent SNodes on the mapped path. In Fig. 5, A and E are adjacent SNodes of failed D. Then, the Dijkstra's algorithm is adopted to find a proper path that connects A and E. The influence weight of a SLink is expressed as

$$D\left(l_{ij}^S\right) = \frac{\gamma}{\left(NoV\%(n_i) + NoV\%(n_j)\right) \cdot TR\left(l_{ij}^S\right)} \tag{21}$$

where γ is a constant set to 1 and $TR\left(l_{ij}^S\right)$ must be higher than $TR\left(l_{um}^V\right)$. $D\left(l_{ij}^S\right)$ presents the node reliability and link

Procedure 3 Node Protection Approach

Input: $G^V = (N^V, L^V)$, $G^S = (N^S, L^S)$, one mapped l^V and corresponding p^S;
Output: l^V protected by p-Cycles;
1. **for** each SNode n of p^S **do**
2. Calculate $NoV\%(n)$ according to Eq. (20);
3. **if** $NoV\%(n) \leq \Gamma_1\%$ **do**
4. Keep n into a node list that is ready for protection in p-Cycle 1;
5. **end if**
6. **end for**
7. **for** each n in the protection list **do**
8. **if** $NoV\%(n) \leq \Gamma_2\%$ **do**
9. Ready to construct a p-Cycle 2;
10. **end if**
11. **if** its adjacent nodes are in the node list **do**
12. Ready to construct a combined p-Cycle;
13. **end if**
14. Construct the p-Cycle by adopting Dijkstra's algorithm with Eq. (21);
15. **end for**

Procedure 4 Recovery From Switching Node Failure

Input: $G^V = (N^V, L^V)$, $G^S = (N^S, L^S)$, l^V and p^S, failed switching node n;
Output: l^V recovered from switching node failure;
1. **if** n is protected by one available p-Cycle **do**
2. Enable the p-Cycle and recover l^V;
3. **else if**
4. Hold the normal SLinks of p^S;
5. Find another proper SNode $\tilde{n}$ to replace n;
6. Find the shortest reliable path that connects $\tilde{n}$ and p^S according to Eq. (21);
7. **end if**

capacity of l_{ij}^S. In Fig. 5, $\{(A,B),(B,E)\}$ is found to construct p-Cycle 1 for D. In some cases, adjacent nodes of D also fail and can be protected by other p-Cycles. To improve resource utilization, these p-Cycles can be combined to protect both n and adjacent nodes. In Fig. 5, the path $\{(A,B),(B,F),(F,I)\}$ can construct a p-Cycle to protect both D and E.

The specific procedures are shown in **Procedure 3** below. The complexity is mainly determined by step 14. The number of SNodes in the protection list is at most $|N^S| - 2$. Also, the complexity of Dijkstra's algorithm is $O(|L^S| + |N^S|\log(|N^S|))$. Thus, the time complexity of **Procedure 3** is $O(|N^S|(|L^S| + |N^S|\log(|N^S|)))$.

2) RECOVERY FROM SWITCHING NODE FAILURE

Procedure 4 presents an efficient and reliable recovery strategy that adopts p-Cycle protection techniques for vulnerable SNodes (step 2) and fast remapping for other switching SNodes (step 6). The complexity is mainly determined by step 2 or 6 which require at most $O(|N^S|(|L^S| + |N^S|\log(|N^S|)))$.

C. ONE-STAGE EMBEDDING AND RECOVERY FROM SINGLE NODE FAILURE

A heuristic one-stage algorithm is first proposed, coordinating node and link mapping to improve embedding profits. Then, we propose a reliable recovery strategy against end or switching node failure.

1) HEURISTIC ONE-STAGE EMBEDDING

The one-stage embedding includes **Procedure 1** and **Procedure 3**, which is presented in **Algorithm 1** below.

Algorithm 1 Embedding in One-Stage

Input: $G^S = (N^S, L^S)$, $G^V = (N^V, L^V)$;
Output: Results of reliably embedding ATVN;
1. $V \leftarrow N^V$; $M \leftarrow \emptyset$;
2. **while** $V \neq \emptyset$ **do**
3. Run ***Procedure 1*** with V as input;
4. Get the output v_u with its SNo_List;
5. $n \leftarrow$ EXTRACT-MAX(SNo_List);
6. Map v_u onto n; $M \leftarrow M \cup \{v_u\}$; $Li \leftarrow \emptyset$;
7. **for** each VNode $v_m \in M$ **do**
8. $Li \leftarrow Li \cup \{l_{um}^V\}$ when $l_{um}^V \in L^V$;
9. **end for**
10. **while** $Li \neq \emptyset$ **do**
11. $l^V \leftarrow$ EXTRACT-MAX(Li);
12. Map l^V by adopting the link mapping approach [32];
13. Run ***Procedure 3*** with l^V as input;
14. Get l^V protected by p-Cycles;
15. **end while**
16. **end while**
17. Update the residual link and nodal resources;

In **Algorithm 1**, we extract the SNode n with the highest ranking value from SNo_List by function EXTRACT-MAX(SNo_List). Li is the set of VLinks which connect unmapped v_u to mapped VNodes. EXTRACT-MAX(Li) extracts the l^V with the highest transmission-rate from Li. The time complexity of **Algorithm 1** is mainly determined by node mapping (step 3), link mapping (step 12) and switching node protection (step 13). The time complexity of step 3, step 12 and step 13 are at most $O(|N^S|^2)$, $O(|L^S| + |N^S|\log(|N^S|))$, and $O(|N^S|(|L^S| + |N^S|\log(|N^S|)))$, respectively. Thus, the total time complexity of **Algorithm 1** is at most $O(|N^V|(|N^S|^2 + |L^V||N^S|(|L^S| + |N^S|\log(|N^S|))))$.

2) RECOVERY STRATEGY AGAINST SINGLE NODE FAILURE

Algorithm 2 combines **Procedure 2** and **Procedure 4** to reliably recover from end and switching node failure.

Each ATVN is assumed to have a predetermined working time according to practical requests of military missions. As single node failure happens, the influences will be worse

Algorithm 2 Recovery From Single Node Failure

Input: $G^S = (N^S, L^S)$, embedded ATVNs, single failed SNode n;
Output: Results of recovering from single node failure;
1. $VN \leftarrow \emptyset$; Record all the failed ATVNs into VN;
2. Sort VN by the remaining working time;
3. **while** $VN \leftarrow \emptyset$ **do**
4. $G^V_{\max} \leftarrow$ EXTRACT-MAX (VN);
5. **if** n maps any VNode v of $G^V_{\max}$ **do**
6. Run ***Procedure 2*** with v as input;
7. **end if**
8. **if** n is a switching SNode of $G^V_{\max}$ **do**
9. Run ***Procedure 4*** with n as input;
10. **end if**
11. **end while**
12. Update the residual link and nodal resources;

TABLE 1. ATSN parameters.

Parameter	Description
Scope	$200km \times 200km$
Number of substrate nodes	50
Number of substrate links	139
Available node power between flight formation	[50, 100], uniform distributed
SLink bandwidth	[20, 50], uniform distributed
SNode Position	[0, 200], x and y coordinates

if remaining working time is longer. To decrease the bad influence, we extract the failed ATVN with longest remaining working time by function EXTRACT-MAX in step 4. The time complexity of ***Algorithm 2*** is mostly determined by step 6 and 9 which require at most $O((|N^S| + |L^V|)(|L^S| + |N^S| \log(|N^S|)))$.

V. SIMULATION EVALUATION

SVNE-FT algorithm is evaluated through extensive simulations. Firstly, we present the simulation setup in detail. Then, we describe the compared algorithms and extensive simulations results.

A. SIMULATION SETTING AND PARAMETERS

We generate the network topology for ATSN and ATVN by an improved Salam network topology random generation algorithm [33]. The Salam network topology generation algorithm can randomly generate corresponding nodes and links that constitute ATSN and ATVN according to the network parameters in Table 1 and Table 2, respectively.

In Table 1, the unit of SNode power is W. The unit of SLink bandwidth is MHz. The power range is based on the practical power range of airborne radio station that includes HF station and VHF station. In Table 2, the unit of VLink transmission rate is $Mbit/s$.

TABLE 2. ATVN parameters.

Parameter	Description
ATVN arrival rate	5 per 100 time units, Poisson process
ATVN lifetime	Average of 1000 time units, exponential distribution
Number of substrate links	[3, 5], uniform distribution
VLink connection probability	0.5
VLink transmission rate	[3, 8], uniform distribution
VNode Position	[0, 200], x and y coordinates

We make some other simulation settings for ATSN and ATVN. Each SNode n is assumed to have enough basic power $p_b(n)$, used within one flight formation. And environmental noise is assumed to be white Gaussian noise with $p_{Noise}(n) = 10^{-6}\ W$. Then, in air-combat field, the complicated link gain needs to consider some parameters (e.g., antenna gains, multipath models) which is not necessary for this paper. Thus, to simplify the analysis process, we set the link gain of a Slink as $G = d^{-k}$, where d is the *Euclidean Distance* between end nodes and k is the channel fading coefficient that is set to be 4. In ATVN, φ^V is set to be 100 km. In Eq. (20), Γ_1 and Γ_2 are set to be 20% and 10%, respectively. The intensity of general interference is denoted by $p_b(n)/AveInf(n) = 1$, which is described as $0dB$ $(S/N = 10\lg(p_b(n)/AveInf(n)))$. Also, it is assumed that malicious attacks from enemy are not too intense. Finally, to simulate single node failure, normal SNodes are ranked into a list by failure value $AveInf(n) + 10^{-4}Rf(n)$, where $Rf(n)$ is a random integer between 1 and 10. We select the SNode with the highest failure value every 2000 time units. Thus, the node failure interval is 2000 time units. The selected SNode will fail and recover after a period that obeys an exponential distribution with a mean of 1000 time units.

The simulations are performed on a Lenovo R720 with Windows 10 operating system, Intel R Core (TM) CPU i7-7700@2.8GHz (8 CPUs) Processor and 8.00G RAM Machine. The simulations run for about 20000 time units on the analysis software MATLAB R2019b that is a continuous event simulator. All the average values of corresponding metrics are obtained by simulating the algorithms for multiple times. In addition, to test the statistical significance of simulation results, the results in all test cases are averaged over 50 runs, and we show the margin of error with a 95% confidence level.

B. COMPARED ALGORITHMS

Table 3 lists out all the heuristic SVNE algorithms with corresponding node protection strategies for experimental simulations. SVNE-FT adopts both reactive and proactive approaches. SVNE-FTs and Fast-ReNoVatE adopts reactive approach. SVNE-ORP adopts proactive approach. Especially, SVNE-ORP and Fast-ReNoVatE are the typical and latest node protection strategies in the literature. SVNE-ORP and

Fast-ReNoVatE, designed for wired networks, may be not proper for SVNE simulations in wireless air-combat field. Thus, they are slightly modified to fit into simulations. Also, to evaluate the performance of p-Cycles, we adopt SVNE-FTs that remaps switching nodes and does not construct p-Cycles for vulnerable switching nodes as SVNE-FT does.

TABLE 3. Compared algorithms.

Notation	Description
SVNE-ORP[34]	End node protection allocates backups for key VNodes and their adjacent VLinks. Switching node protection is not taken into account.
Fast-ReNoVatE[17]	End node protection remaps the affected VNodes and adjacent VLinks by computing the maximum flow. Switching node protection remaps the whole affected VLinks by finding the shortest path.
SVNE-FTs	End node protection adopts the ***Procedure 1*** proposed in this paper. Switching node protection remaps failed switching SNodes and adjacent SLinks for the affected VLinks, which is proposed in this paper.
SVNE-FT	End node protection adopts ***Procedure 1*** in this paper. Switching node protection adopts ***Procedure 3*** and ***4*** in this paper.

C. SIMULATION RESULTS

In this sub-section, we discuss the simulation results in terms of average acceptance ratio, longterm average revenue to cost ratio, average execution time, average failure recovery ratio and average failure recovery delay.

1) APPARENT BASIC PERFORMANCE ADVANTAGE

Fig. 6 shows two conclusions. Firstly, average acceptance ratios decrease quickly in the beginning simulation stage and reach stable states in the post simulation stage. This is because there is a balance between the available network resource of ATSN and the number of accepted ATVNs that cost resources. Also, the available network resource will decrease as SNodes and SLinks fail. Secondly, SVNE-FT has the highest average acceptance ratio about 0.65 as simulation time extends. In SVNE-FT, **Procedure 2** is adopted to efficiently protect VNodes against end node failure, decreasing resource cost. **Procedure 3** and **4** are adopted to protect VLinks against switching node failure, increasing recovery efficiency. In comparison with SVNE-FT, SVNE-FTs does not construct p-Cycles for vulnerable switching SNodes and thus may not recover more VLinks from switching node failure. Fast-ReNoVatE remaps failed VLinks, by releasing the whole substrate path, decreasing recovery efficiency. SVNE-ORP allocates backups for VNodes and VLinks that may be affected by end node failure, costing more network resource. Also, it does not consider the recovery of the failed VLinks affected by switching node failure.

Fig. 7 illustrates the comparison results of longterm average revenue to cost ratios. As simulation time extends, longterm average revenue to cost ratios will decrease to stable states. SVNE-ORP has the lowest ratio because it allocates backups for key VNodes and VLinks, increasing resource

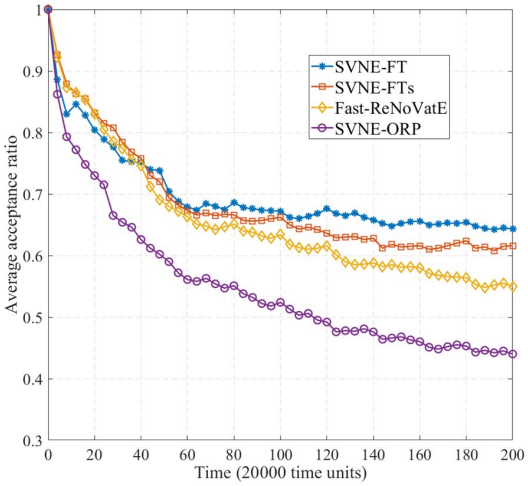


FIGURE 6. Average acceptance ratio.

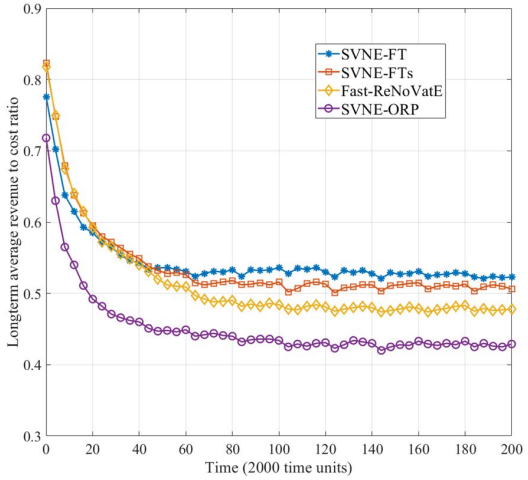


FIGURE 7. Longterm average revenue to cost ratio.

cost and the average length of paths that map VLinks. In comparison with SVNE-ORP, Fast-ReNoVatE remaps failed VNodes and VLinks, saving extra network resource and increasing embedding revenue. However, SVNE-FT and SVNE-FTs behave better due to the recovery strategy against single node failure, adopting **Procedure 2**. To end node failure, they just release part of the path and thus increase the embedding revenue of ATVNs. To switching node failure, SVNE-FTs remaps the failed VLinks according to Eq. (21). SVNE-FT adopts **Procedure 3** and **4** to protect failed VLinks, increasing the recovery efficiency and embedding revenue. Through the simulation, SVNE-FT further ensures the efficiency of substrate network resource in the long run. Thus, it has the highest longterm average revenue to cost ratio about 0.52.

Fig. 8 illustrates the average execution time of the compared algorithms as a function of time. As simulation time extends, the limited available network resource may cause the increase of average execution time. But it will increase to a stable state due to the balance between available network resource and accepted ATVNs. Firstly, to SVNE-ORP, it has

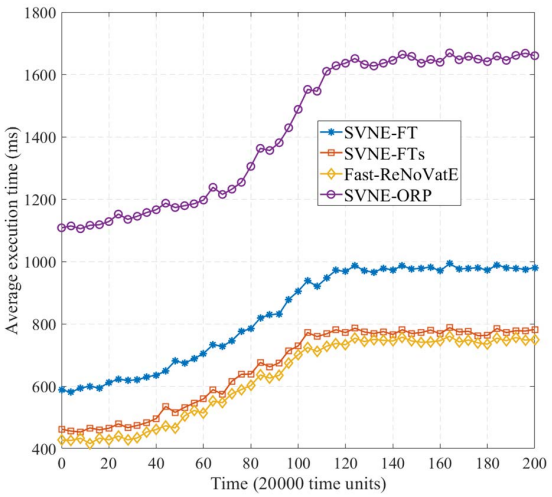


FIGURE 8. Average execution time.

the highest average execution time during the simulation process. The reason is that SVNE-ORP requires to find backups for most of VNodes and corresponding VLinks, which may spend more time embedding ATVNs. Then, to SVNE-FT, it only requires to construct p-Cycles for part of SNodes on the substrate path that maps corresponding VLinks, spending less time embedding ATVNs. Finally, to SVNE-FTs and Fast-ReNoVatE, the two reactive heuristic SVNE algorithms don't require to prepare backups for any VNode or VLink and have less average execution time.

2) RECOVERY PERFORMANCE IMPROVEMENT

Fig. 9 depicts the average failure recovery ratio as a function of time. As the simulation time extends, the ratio will consistently decrease due to the fact that there is less available network resource to protect ATVNs against single node failure. Firstly, SVNE-ORP has a higher ratio in the beginning simulation stage with enough resources to allocate backups against single node failure. However, it has the lowest ratio in the post simulation stage with less resources to allocate backups. It also pays no attention to the protection of failed VLinks from switching node failure. In contrast, Fast-ReNoVatE performs better because it accounts for the switching node failure and remaps the failed VNodes and VLinks, decreasing the resource cost in the post simulation stage. Obviously, SVNE-FT and SVNE-FTs have an apparent advantage in the post simulation stage. To end node failure, they adopt **Procedure 2** to find more proper SNodes and SLinks for failed VNodes. Also, to switching node failure, they release part of the substrate path and can find reliable SNodes and SLinks. Thus, more failed ATVNs can be recovered from single node failure. Compared with SVNE-FTs, SVNE-FT adopts **Procedure 3** and **4**, constructing p-Cycles to protect vulnerable SNodes against switching node failure. Therefore, SVNE-FT has the highest average failure recovery ratio as time extends.

Fig. 10 shows the average failure recovery delay of the compare algorithms as a function of time. As the simulation time extends, the average failure recovery delay increases to a stable state, which is probably due to all the embedding algorithms need to take time to find enough network resource to remap VNodes or VLinks after single node failure happens. In addition, SVNE-ORP has the lowest average failure recovery delay about 12.5ms due to its backups prepared for all key VNodes. To our proposed SVNE-FT, it has the average failure recovery delay about 13.3ms that is lower than both SVNE-FTs and Fast-ReNoVatE because it prepares backups for unreliable switching nodes. SVNE-ORP has a delay that is just about 6% lower than that of SVNE-FT. However, SVNE-FT has an average failure recovery ratio that is about 20% higher than that of SVNE-ORP. Thus, SVNE-FT has better recovery performance.

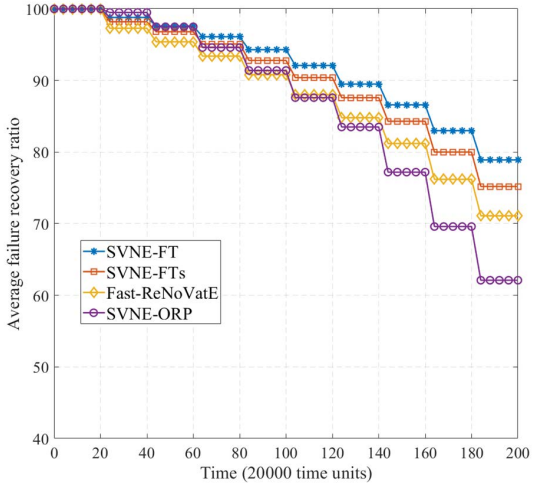


FIGURE 9. Average failure recovery ratio.

3) PERFORMANCE CHANGE UNDER THE CHANGES OF KEY VARIABLES.

Fig. 11 illustrates the stable states of average acceptance ratio under different node failure interval. For example, as node failure interval is 2000 time units, the final stable state of average acceptance ratio is about 0.65 for SVNE-FT as shown in Fig. 6. As shown in Fig. 10, the average acceptance ratio increases as the node failure interval increases, which is probably due to the fact that there may be less node failures as the node failure interval is bigger. In case of our proposed SVNE-FT algorithm, the average acceptance ratio increases from 0.21 to nearly 0.71 as the node failure interval increases from 500 to 2500 time units. In addition, the average acceptance ratios of other three algorithms are lower than SVNE-FT as the interval increases. The major reason of the higher ratio in case of SVNE-FT is due to the better protection against end node or switching node failure.

Fig. 12 shows the stable states of average acceptance ratios under different Γ_1 and Γ_2 in SVNE-FT. The average acceptance ratio under Γ_1 denotes the stable state of average acceptance ratio under the condition that a switching node is protected only by p-Cycle 1 if its $NoV\%\,(n)$ is less than Γ_1. Similarly, the average acceptance ratio under Γ_2 denotes the stable state of average acceptance ratio with switching node

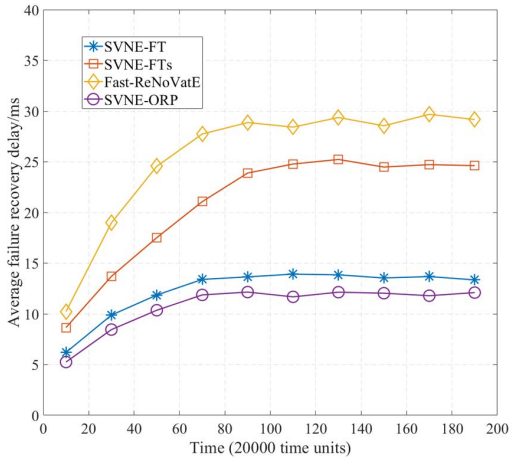


FIGURE 10. Average failure recovery delay.

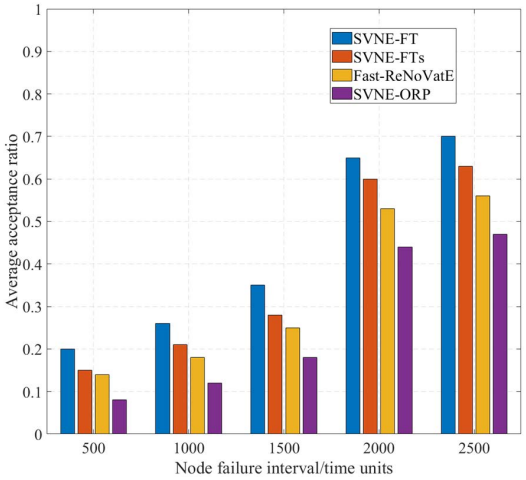


FIGURE 11. Average acceptance ratio under different node failure interval.

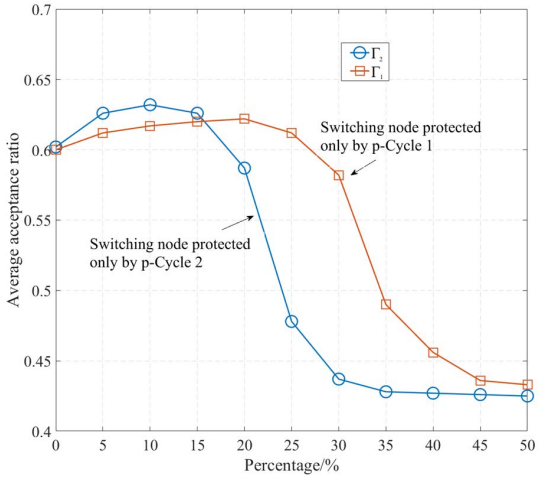


FIGURE 12. Average acceptance ratios under different Γ_1 and Γ_2.

protected only by p-Cycle 2 if its $NoV\%$ (n) is less than Γ_2. As shown in Fig. 10, the average acceptance ratios under different Γ_1 and Γ_2 are graphs of concave functions. This is probably due to fact there is a resource balance between resources used by back-up p-Cycles and resources used by embedding new VNs. In addition, the average acceptance ratio under different Γ_2 will reach a maximum about 0.63 as Γ_2 is equal to be about 10% while the ratio under different Γ_1 will reach a maximum about 0.62 as Γ_1 is equal to be about 20%. Thus, we set $\Gamma_2 = 10\%$ and $\Gamma_1 = 20\%$ to get comparably higher average acceptance ratio of the proposed SVNE-FT algorithm as shown in Fig. 6.

VI. CONCLUSION

In this paper, we have addressed the problem of recovering failed ATVNs from a single substrate node failure (end and switching node failure) in the wireless airborne field.

To tackle the computational complexity, we have presented a heuristic algorithm, SVNE-FT. To end node failure, SVNE-FT is able to remap the failed VNodes and adjacent VLinks by releasing part of substrate paths. To switching node failure, SVNE-FT efficiently constructs p-Cycles to protect the affected VLinks. To decrease the resource cost, it selects the vulnerable SNodes of the path and constructs two types of p-Cycles. Different simulation results validate that SVNE-FT algorithm outperforms typical and latest heuristic SVNE algorithms in the wireless environment of ATNs. Main performance metrics are plotted, such as average acceptance ratio, longterm average revenue to cost ratio, average failure recovery ratio and average execution time.

In the future, there are still a number of issues to be done. First of all, this work is extended to recover from SLink failures, resulting from over-subscribed bandwidth allocation. Next, it is to study the problem in a real testbed environment and evaluate the SVNE-FT algorithm through a prototype implementation.

ACKNOWLEDGMENT

The authors would like to thank the editors and the anonymous reviewers whose insightful comments have helped to improve the quality of this paper considerably.

REFERENCES

[1] N. B. Gaikwad, H. Ugale, A. Keskar, and N. C. Shivaprakash, "The internet-of-battlefield-things (IoBT)-based enemy localization using soldiers location and gunshot direction," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11725–11734, Dec. 2020.

[2] P. Zhang, C. Wang, N. Kumar, and L. Liu, "Space-air-ground integrated multi-domain network resource orchestration based on virtual network architecture: A DRL method," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2798–2808, Mar. 2022.

[3] B. N. Cheng, "Design considerations for next-generation airborne tactical networks," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 138–145, May 2014.

[4] J. Wang, P. Deutsch, A. Coyle, T. Shake, and B.-N. Cheng, "An implementation of a flexible topology management system for aerial high capacity directional networks," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2015, pp. 991–996.

[5] M. Jingcheng, L. Na, C. Kefan, C. Zhuo, and G. Weiting, "A highly reliable embedding algorithm for airborne tactical network virtualization," *J. Syst. Eng. Electron.*, vol. 32, no. 6, pp. 1364–1374, Dec. 2021.

[6] K. Chen, S. Zhao, N. Lv, W. Gao, X. Wang, and X. Zou, "Segment routing based traffic scheduling for the software-defined airborne backbone network," *IEEE Access*, vol. 7, pp. 106162–106178, 2019.

[7] X. Zhang and Q. Zhu, "Game-theory based power and spectrum virtualization for optimizing spectrum efficiency in mobile cloud-computing wireless networks," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1025–1038, Oct. 2019.

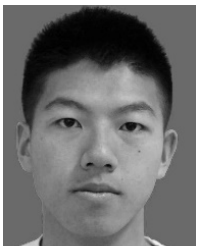
[8] N. N. Sapavath and D. B. Rawat, "Wireless virtualization architecture: Wireless networking for Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5946–5953, Jul. 2020.
[9] J. van de Belt, H. Ahmadi, and L. E. Doyle, "Defining and surveying wireless link virtualization and wireless network virtualization," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1603–1627, 3rd Quart., 2017.
[10] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 655–685, Jan. 2016.
[11] H. Cao, H. Hu, Z. Qu, and L. Yang, "Heuristic solutions of virtual network embedding: A survey," *China Commun.*, vol. 15, no. 3, pp. 186–219, Mar. 2018.
[12] 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.
[13] X. Liu and D. Medhi, "Optimally selecting standby virtual routers for node failures in a virtual network environment," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 2, pp. 275–288, Jun. 2017.
[14] W. Hou, Z. Ning, and L. Guo, "Green survivable collaborative edge computing in smart cities," *IEEE Trans. Ind. Informat.,* vol. 14, no. 4, pp. 1594–1605, Apr. 2018.
[15] H. Cao, Y. Hu, Q. Wang, S. Wu, and L. Yang, "A blockchain-based virtual network embedding algorithm for secure software defined networking," in *Proc. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Jul. 2020, pp. 1057–1062.
[16] B. Lu, T. Huang, X. Sun, J.-Y. Chen, and Y.-J. Liu, "Dynamic recovery for survivable virtual network embedding," *J. China Univ. Posts Telecommun.*, vol. 21, no. 3, pp. 77–84, Jun. 2014.
[17] N. Shahriar, R. Ahmed, S. R. Chowdhury, A. Khan, R. Boutaba, and J. Mitra, "Generalized recovery from node failure in virtual network embedding," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 2, pp. 261–274, Jun. 2017.
[18] M. Li, C. Chen, C. Hua, and X. Guan, "Intelligent latency-aware virtual network embedding for industrial wireless networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7484–7496, Oct. 2019.
[19] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Efficient virtual network embedding with backtrack avoidance for dynamic wireless networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2669–2683, Apr. 2016.
[20] N. Pathak, S. Misra, A. Mukherjee, A. Roy, and A. Y. Zomaya, "UAV virtualization for enabling heterogeneous and persistent UAV-as-a-service," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6731–6738, Jun. 2020.
[21] H. Mei, K. Yang, Q. Liu, and K. Wang, "Joint trajectory-resource optimization in UAV-enabled edge-cloud system with virtualized mobile clone," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5906–5921, Jul. 2020.
[22] S. Li, M. Y. Saidi, and K. Chen, "Survivable services oriented protection level-aware virtual network embedding," *Comput. Commun.*, vol. 152, pp. 34–45, Feb. 2020.
[23] A. Xiao, Y. Wang, L. Meng, X. Qiu, and W. Li, "Topology-aware virtual network embedding to survive multiple node failures," in *Proc. IEEE Global Commun. Conf.*, Austin, TX, USA, Dec. 2014, pp. 1823–1828.
[24] X. Chang, J. K. Muppala, B. Wang, J. Liu, and L. Sun, "Migration cost aware virtual network re-embedding in presence of resource failures," in *Proc. 18th IEEE Int. Conf. Netw. (ICON)*, Singapore, Dec. 2012, pp. 24–29.
[25] A. Jarray, Y. Song, and A. Karmouch, "P-cycle-based node failure protection for survivable virtual network embedding," in *Proc. IFIP Netw. Conf.*, Brooklyn, NY, USA, 2013, pp. 1–9.
[26] 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.
[27] K. Chen, N. Lv, and S. Zhao, "A scheme for improving the communications efficiency between the control plane and data plane of the SDN-enabled airborne tactical network," *IEEE Access*, vol. 6, pp. 37286–37301, 2018.
[28] Z. Li and Y. Wu, "Smooth mobility and link reliability-based optimized link state routing scheme for MANETs," *IEEE Commun. Lett.*, vol. 21, no. 7, pp. 1529–1532, Jul. 2017.
[29] C. Liang and F. R. Yu, "Wireless network virtualization: A survey, some research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 358–380, Mar. 2015.
[30] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos, "Wireless sensor network virtualization: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 553–576, 1st Quart., 2016.
[31] Y. Miao, C. Wu, and Q. Yang, "Self-healing mechanism for reconfigurable service overlay networks," *J. Commun.*, vol. 33, no. 8, pp. 52–61, 2012.
[32] J. Miao, N. Lv, K. Chen, W. Gao, and Y. Zhang, "A reliable interference-aware mapping algorithm for airborne tactical network virtualization," *IEEE Access*, vol. 9, pp. 5083–5096, 2021.
[33] Z. Zhao, "Virtual network embedding based on node connectivity awareness and path integration evaluation," *KSII Trans. Internet Inf. Syst.*, vol. 11, no. 7, pp. 3393–3412, 2017.
[34] W. Yeow, C. Westphal, and U. C. Kozat, "Designing and embedding reliable virtual infrastructures," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 2, pp. 57–64, Apr. 2011.

JINGCHENG MIAO received the B.S. degree in telecommunication engineering and the M.S. degree in signal and information processing from the Engineering University of PAP, Xi'an, China, in 2016 and 2018, respectively. He is currently pursuing the Ph.D. degree in aerospace network with Air Force Engineering University, Xi'an.

His research interests include airborne tactical networks, network virtualization, and virtual network embedding.

NA LV received the B.S. degree in testing technology and instrumentation, the M.S. degree in control theory and applications, and the Ph.D. degree in armament science and technology from Northwestern Polytechnical University (NWPU), Xi'an, China, in 1992, 1995, and 2010, respectively.

She is currently a Full Professor with Air Force Engineering University, Xi'an. Her current research interests include aviation data link systems and wireless network virtualization.



QI GAO received the B.S. degree in telecommunication engineering from Air Force Engineering University, Xi'an, China, where he is currently pursuing the M.S. degree in communication and information system.

His research interests include network virtualization, machine learning, and military aviation communication.

KEFAN CHEN received the B.S. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2013, and the M.S. and Ph.D. degrees from Air Force Engineering University, Xi'an, China, in 2016 and 2019, respectively.

He is currently an Engineer. His research interests include airborne tactical networks, software-defined networking, and network virtualization.

XIANG WANG received the B.S., M.S., and Ph.D. degrees in communication and information system from Air Force Engineering University, Xi'an, China, in 2006, 2009, and 2013, respectively.

He is currently a Lecturer with Air Force Engineering University. His research interests include airborne networks, wireless network virtualization, and software-defined networking.