Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7164854, 19 pages
https://doi.org/10.1155/2022/7164854

Research Article

Dynamic Reliability-Aware Virtual Network Embedding for Airborne Tactical Networks

Jingcheng Miao, Na Lv, Kefan Chen, Qi Gao, and Xiang Wang

School of Information and Navigation, Air Force Engineering University, Xi'an 710077, China

Correspondence should be addressed to Na Lv; lvnn2007@163.com

Received 26 April 2022; Revised 25 August 2022; Accepted 3 September 2022; Published 16 September 2022

Academic Editor: Manuel Fernandez-Veiga

Airborne tactical networks (ATNs) built on network virtualization (NV) can enable efficient information sharing for network-centric warfare by breaking the tight coupling between applications and network infrastructure and thus solving the network ossification problem. With dynamic changes during military missions, the application of virtualization is challenged by the changing demands on network resources when instantiating multiple virtual networks (VNs) on a shared substrate network (SN), known as virtual network embedding (VNE). However, existing dynamic VNE algorithms, mostly designed for wired networks, focus on the dynamic changes of single VN and do not consider the dynamic changes of different VNs. The proper processing of changes of different VNs is beneficial to improve the embedding profits of VNs. On these backgrounds, we investigate the dynamic wireless VNE that processes the dynamic changes of different VNs in the wireless environment of ATNs. A relationship matrix-based dynamic wireless VNE (DWVNE-RM) algorithm is proposed in order to maximize the acceptance ratio of VN requests. The dynamic changes of different VNs are classified into negative changes (requiring resource) and positive changes (releasing resource). Then, the relationship matrix between negative and positive changes is constructed to properly process the resource relationship among different changes in order to improve the acceptance ratio. In addition, the complex interference of ATNs is considered in the mapping or remapping of virtual nodes and links to improve the reliability. Through extensive simulation, DWVNE-RM is compared with multiple existing dynamic algorithms and outperforms over others.

1. Introduction

Airborne tactical networks (ATNs) [1] exploit manned or unmanned aircrafts with various warfare capabilities of aviation communication, which are being reshaped by the revolution of mosaic warfare [2] and Internet of Battle Things [3]. ATNs have recently gained significant attention due to the communication demands for rapidly deploying network resources to maintain coverage and provide reach-back to aircrafts as well as surface and space military units [4, 5]. These communication demands are characterized by dynamic resource requirement due to the dynamic changes during the missions. Unfortunately, the provision of rapid adaption to changes is still a challenging issue due to the insufficient cross-platform interoperability in the ossification problem of current ATNs [6, 7]. The ossification problem refers to the fact that current ATNs are application coupled with vertical integration of customized software stack and hardware [8], impeding the cross-platform utilization of network resources. Therefore, abstracting network resources is compelling to support multi-mission multi-applications and improve the adaption to mission changes for ATNs.

To solve the network ossification problem, network virtualization (NV) is evolving as a major technological breakthrough in many emerging areas like cloud computing [9] and smart IoT [10]. It abstracts network resources and allows network users to program network services as multiple virtual networks (VNs) that are built and run on the same substrate network (SN) [11, 12]. Thus, implementing NV can provide ATNs a high level of resource sharing among various air-combat applications. In addition, software defined networking (SDN) can offer an ideal platform for implementing NV because it enables strict quality of service (QoS) provisioning without tunneling overhead in comparison with conventional virtualization methods [13]. Thus, a viable virtualization platform can be provisioned for ATNs, called as airborne tactical network virtualization (ATNV), by using SDN to implement NV.

Virtual network embedding (VNE) is a vital step to realize ATNV. It can deal with the resource allocation for mission applications that are abstracted into VNs with customized resource demands (e.g., power, transmission rate, and delay) on corresponding virtual nodes and links. To solve the NP-hard VNE problem, many heuristic algorithms (e.g., game theory [14] and reinforcement learning [15]) have been proposed in the literature [14–22]. However, these VNE algorithms are mostly designed for wired networks. Due to the broadcast nature of the wireless channel, the wireless VNE problem is a new undertaking that considers co-channel interference and noise, especially for ATNV.

In ATNV, there are specific two constraints into the VNE. First, in ATNs, mission applications are influenced by both complex electromagnetic environment and electronic warfare which constrain the mapping of virtual nodes and links onto substrate nodes and paths. However, existing wireless VNE algorithms mostly focus on the constraints on the mapping of virtual links [23–26], which consider either multi-hop link instability or the co-channel interference between neighboring nodes. Due to the close relationship between node mapping and link mapping, it is essential to take the link interference into account when mapping virtual nodes. Instead, a reliable mapping approach scheme was proposed under complex interference in our previous work [27]. It adopts the Shannon formula to rank all the valuable physical nodes for mapping virtual nodes. Second, the QoS-compliant capacity of mission applications is challenged by the rapid changes of the missions in ATNs. For example, the command and control center requires to handle large amount of batch and streaming data from the complex air-combat environment. As the collected situation information and data changes, mission applications may generate different resource demands. Thus, it is needed for ATNV to allow the changes in the structure of VNs for mission applications to handle the growth of data. Although some dynamic VNE algorithms have been proposed [28–30], they focus on processing dynamic changes of single VN and neglect the complementary resource relationship among dynamic changes of different VNs. In addition, it is practically found that there may be not enough resource of substrate network to successfully process dynamic changes due to the fact that the resource is mostly used by embedded VNs and newly arrived VNs. Adopting the complementary relationship among changes can release relevant resource to properly process the dynamic changes and thus reaccept the dynamic VNs, improving the acceptance ratio.

In this context, this paper investigates the dynamic wireless VNE due to the changes in the resource requirement and network topology of different VNs. First, for newly arrived VN requests, virtual nodes and links of VNs are reliably mapped onto substrate nodes and paths, respectively, considering the complex interference. Next, for the accepted VNs with dynamic changes, the changes are divided into negative changes and positive changes. The relationship matrix is constructed to process the resource relationship between negative and positive changes, improving the acceptance ratio. Finally, simulation results are assisted to validate the strength of our heuristic dynamic VNE algorithm for ATNV through extensive simulation. Performance metrics (e.g., average acceptance ratio) are illustrated to highlight the proposed algorithm.

The main contributions of this paper are summarized as follows:

(1) A novel relationship matrix-based dynamic wireless VNE algorithm is proposed to improve the acceptance ratio, considering the resource relationship among dynamic changes. By doing so, SDN controllers are allowed to change the structure and resource configuration of the entire VNs for the corresponding mission applications

(2) The transmission rate of each physical path is used during the mapping process of virtual links to achieve a balance between resource availability and complex interference. In addition, the transmission rate of each virtual link is also used to describe the resource relationship among dynamic changes of different VNs

(3) Numerical simulations are performed in this paper. The proposed DWVNE-RM algorithm is compared with multiple existing dynamic VNE algorithms. In addition, simulation results are illustrated in order to highlight the proposed algorithm

The rest of the paper is organized as follows. Section 2 presents the related works. In Section 3, we present the SDN based model for ATNV and formulate the dynamic VNE problem. Then, Section 4 presents the details of the proposed DWVNE-RM algorithm. The simulation work is implemented in Section 5, along with parameters setting. Finally, Section 6 concludes our findings.

2. Related Works

ATNs are currently constructed on discrete data link systems (e.g., Link-16, TTNT, and MADL) [8]. 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 fulfill the actual desire to support an ever-increasing number of users and emerging applications and then adapt to the dynamic changes during missions [1]. 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 [31, 32]. Therefore, NV can improve the adaption to changes during missions for ATNs, which is based on the VNE to allocate virtualized network resources efficiently.

2.1. Previous Typical and Latest Works. Most of prior VNE algorithms are static and designed for wired networks [14–22], which adopt game theory [14], reinforcement learning (RL) [15], graph theory [20, 21], and so on. Authors in [21] proposed two heuristic layered-auxiliary-graph algorithms to get proper matching between VNs and SN in the optical network virtualization. Also, authors in [22] considered the availabilities of both substrate nodes and links to solve the survivable VNE problems with network failures. However, these works are not proper for network virtualization in wireless networks for they do not consider wireless link interference.

Recently, researchers have paid more attention to wireless VNE of which the main problem is to deal with the link interference. Therefore, authors in [24] proposed a shortest-path based approach that defined interference weights for wireless links in a one-hop model, and authors in [25] considered wireless interference by searching through all possible embeddings to explore the trade-off between solution quality and search time. Different from single or multiple shortest-path approach, an anypath link mapping scheme was proposed in [23] to deal with the multi-hop link instability. Nevertheless, these works do not fully consider the influence of link interference when mapping virtual nodes, which can provide high reliability for the virtualization of ATNs. Therefore, a reliable mapping approach scheme was proposed for ATN virtualization under harsh air-combat field in our previous work [27]. It adopted the Shannon formula to estimate the reliability of wireless nodes for mapping virtual nodes.

Moreover, in recent years, there are many dynamic VNE approaches in network virtualization. Then, we will detail typical and latest dynamic VNE algorithms mainly from two aspects.

On the one hand, some dynamic mapping strategies processed the reconfiguration issues in the assumption that the network status of SN would change as the VN requests existed. In [33], authors proposed a dynamic and collaborative mapping mechanism for a multilayer VNE framework in SDN. If the under VN failed to satisfy the resource demands of up VN, this mechanism adopted a reinforcement learning based approach to increase the resource capacity of virtual elements of under VN or migrating them to other physical nodes or paths of SN. This kind of dynamic VNE algorithm in the multilayer VNE framework is out of our scope. In [30], authors proposed a dynamic heuristic algorithm to be evaluated in continuous time. To the embedded VN request, the approach would drive the re-embedding scheme and adjust certain virtual elements if the QoS demand of this VN was not guaranteed. However, it is designed for wired networks which is not optimal in the complex wireless environment of ATNs. For wireless industrial networks, authors in [34] proposed a learning-based dynamic anypath mapping scheme to provide intelligent monitoring and treatments in the SDN-based framework. It focused on the time-varying link quality and dynamic node workload. Though of interest, the on-site timely forwarding adjustments for per VN was time-consuming.

On the other hand, researchers considered the re-embedding process of VN request due to the changes in resource demands and traffic. In [35], authors first addressed the issue of how to reconfigure and map an existing VN request, while its topology and resource requirement changed as the customer's requirements changed. A heuristic dynamic mapping was proposed to deal with the VN evolution including four scenarios which were add components, delete components, decrease resource requirements, and increase resource requirements. In [36], the dynamics of virtual network demands were modelled as a combination of a Gaussian distribution and a daily diurnal pattern. Heuristic dynamic algorithms were designed to minimize the energy consumption of substrate network. Also, authors in [37] considered that VNs had periodic demands sometimes and thus reassigned unused bandwidth efficiently. Moreover, the authors in [29] considered the resource demand of the VN and the resource availability of the SN at different time instances. To maximize the resource utilization, they proposed a fitness-based heuristic mapping algorithm to improve the resource utilization by decreasing the remaining idle resource. However, these mapping algorithms are designed for wired networks and ignored the resource relationship among changes of different VN requests.

Finally, there are also some dynamic virtual network function deployment algorithms (e.g. [38–40]) in network function virtualization. For example, authors in [38] focused on dynamic readjustment of service function chains with changing demands of network users. However, these works, mostly designed for wired networks, focus on the dynamic change of one service function chain and do not consider the relationship among dynamic changes of different service function chains.

2.2. Brief Summary of Previous Works. To adapt well to the changes during missions, NV can be exploited to solve the ossification problem of ATNs. As one key technical issue of NV, dynamic VNE can embed the VNs and thus fulfil the resource demands of the changes during missions. However, the existing dynamic VNE algorithms fail to consider the resource relationship among changes of different VNs, which can improve the embedding profits.

Therefore, we investigate the dynamic VNE problem to improve the adaption to changes during missions for the virtualization of ATNs. The technical issues of dynamic VNE have not been fully studied in the academia yet [30]. Due to the NP-hard nature of VNE problem, it is not practical to develop pure reinforcement learning based algorithms and exact algorithms. Hence, we propose a heuristic algorithm that can be implemented and evaluated in continuous time. Different from previous dynamic algorithms for wired [29, 33] and wireless [34] networks, we consider the resource relationship among changes of different VN requests to improve the embedding profits. In addition, the link interference is considered while mapping virtual nodes and virtual links in wireless airborne field.

3. Problem Formulation

In this section, the network model of ATNV and dynamic VNE are presented. Afterward, the relationship matrix and relationship value are formulated, followed by the formulation of the objective function.

3.1. Slice-Based ATNV Model. ATNs, with the ossification problem, are constituted of massive manned or unmanned aircrafts as well as surface and ground platforms in vertically integrated network architecture. To solve the ossification problem, it is pertinent to design a scalable and flexible infrastructure for ATNs. Inspired by the SDN-enabled ATN [41] and the virtualization scheme [34], we design a slice-based ATNV framework in Figure 1.

3.1.1. Substrate Network Layer. Massive aircrafts are deployed in concomitant formation including lead aircraft and wingman. Lead aircrafts, equipped with SDN-enable devices, form a large-scale wireless multi-hop network that can be abstracted as the shared airborne tactical substrate network (ATSN).

3.1.2. Virtualization Layer. A hypervisor abstracts the network resource of ATSN in service for mission applications. The cloud icon represents the tactical data center that owns strong information processing capability to deal with the rapid changes of situation information and mission applications. Thus, it can adopt the dynamic VNE algorithm and orchestrate multiple applications in an overlapped way.

3.1.3. Application Layer. Mission applications are precisely defined in logical and self-contained airborne tactical virtual networks (ATVNs) with dedicated SDN controllers deployed to support the protocol reconfiguration and evolution. Also, ATVNs are allowed to change topologies and customized QoS demands.

In Figure 1, we illustrate 3 mission applications with strict transmission rate request. For instance, ATVN 1 has reconnaissance mission and contains distributed aircraft 2 to 5 and access point 1 to cloud devices. It is embedded onto ATSN by the hypervisor and then collects real-time situation information for operational decision and further military missions.

All aircrafts can be equipped with discrete data link systems (e.g., Link-16, TTNT, and MADL) [42] which allow aircrafts to transmit information about its current status (e.g., geographical position updates, intents, and speed) periodically and automatically. Thus, in most cases, the hypervisor in application layer can obtain the topology of ATSN formed by lead aircrafts, timely and accurately [43].

However, part of the topology sometimes cannot be obtained timely due to the fact that airborne tactical networks can be influenced by the movement of the aircraft, the orbit the aircraft flies in, the placement of antennas, and so on. In this case, multiple technologies (e.g., delay tolerant networking [44] and optimized link state routing [45]) can be adopted to deal with this problem. For example, delay tolerant networking can be adopted to ensure data transmission by storing the data at intermediary nodes during outages. Moreover, the unavailable links will influence the link mapping of virtual links in the VNE process. We have considered this problem in our previous work [26], which selects multiple reliable paths to map virtual links by opportunistic routing with differentiated transmission rates.

In general, four main features exist in the slice-based ATNV. First of all, a software designed platform is provided to allow mission applications to neglect the technical details of ATNs and flexibly adapt to rapid changes [11]. Second, the hypervisor has global state view in resource usage and can efficiently allocate resource to satisfy the changing demands of mission applications. Third, various ATVNs are allowed to evolve their own protocols, which ensures full isolation and improves the efficiency of dynamic network management [46]. Finally, it owns high scalability and sustainability, which can decrease the overhead of extra devices to develop innovative applications and thus accelerate the deployment speed of new network technologies for the changing demands [12].

3.2. Dynamic VNE Problem Model

3.2.1. ATSN and ATVN. As shown in Figure 1, lead aircrafts form a wireless multi-hop network that can be abstracted as ATSN. In Figure 2, we represent it as an undirected graph $G^S = (N^S, L^S)$, where N^S is the set of substrate nodes (SNodes), L^S is the set of substrate links (SLinks), and P^S is the set of substrate paths. Each SNode $n \in N^S$ is associated with mainly two parameters that are radio transmission power tp^S and location $loc(n)$. In this paper, the location is defined by x and y coordinates. tp^S includes basic power $tp_b(n)$ and cooperation power $tp(n)$, which are used within and between aviation formations, respectively. Each SLink $l^S \in L^S$ is associated with bandwidth capacity $b(l^S)$ and relative location $loc(l^S)$. Each path $p^S_{ij} \in P^S$ is loop-free substrate path between SNode n_i and n_j.

Figure 1 also shows three ATVNs that are abstracted from mission applications. Since this paper deals with the dynamic nature of missions, we design the ATVN as a function of time. As a result, an ATVN is represented by an undirected graph $G^V_t = (N^V_t, L^V_t)$, where N^V_t and L^V_t represent the set of virtual nodes (VNodes) and virtual links (VLinks) at time t, respectively. Each VNode $v_t \in N^V_t$ is associated with location $loc(v_t)$ and coverage radius φ^V. Each VLink $l^V_{um,t} \in L^V_t$ is associated with transmission rate $TR(l^V_{um,t})$ and end VNodes $v_{u,t}$ and $v_{m,t}$.

As shown in Figure 2, two ATVNs with different topologies are embedded onto the shared ATSN. $TR(l^V_t)$ is denoted by the number over VLinks of ATVNs. In ATSN, the number beside SNodes denotes $tp(n)$. And the number over SLinks denotes $b(l^S)$. The embedding process includes node and link mapping. Node mapping is represented by the mapping function M^N: $v_t \in N^V_t \longrightarrow n \in N^S$. For example, node mapping results of ATVN 1 are $M^N(x) = B$, $M^N(y) = C$, and $M^N(z) = G$. Link mapping is represented by another function M^L: $l^V_t \in L^V_t \longrightarrow p^S_{ij} \in P^S$. Also, link mapping results of ATVN 1 are $M^L((x, y)) = \{(B, D), (D, C)\}$ and $M^L((x, z)) = \{(B, F), (F, E), (E, G)\}$.

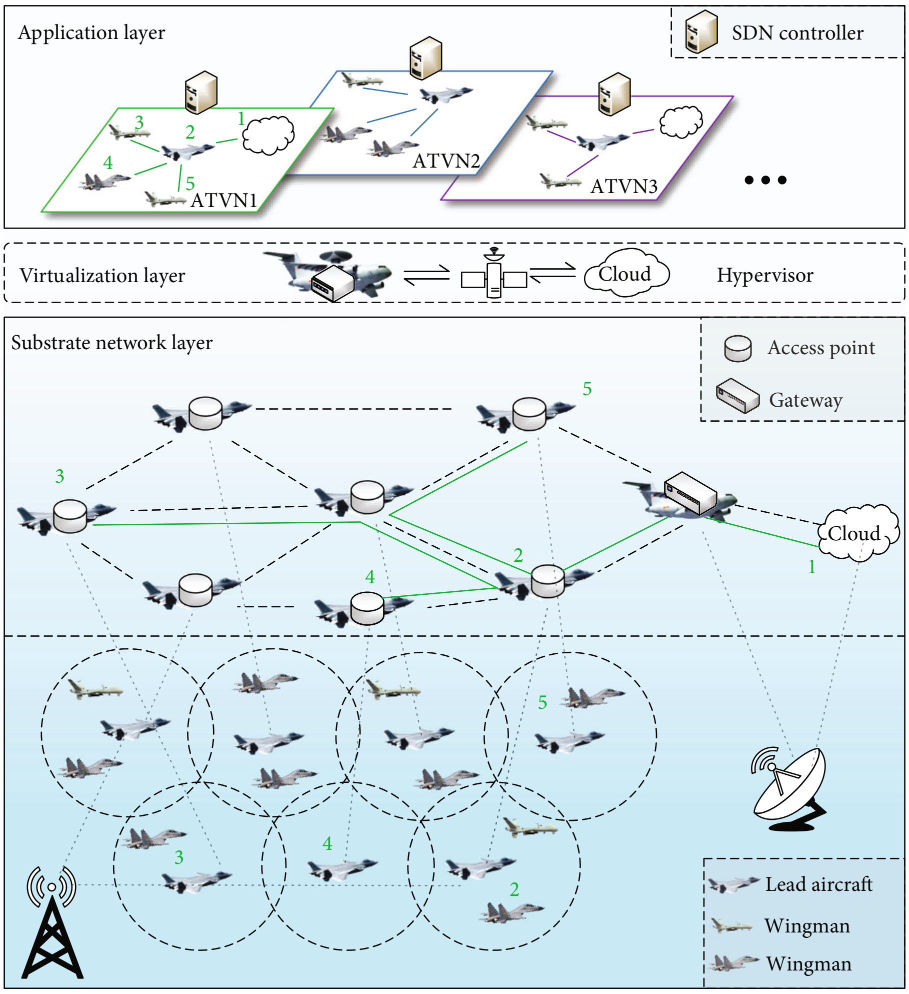


FIGURE 1: Illustration of slice-based ATNV.

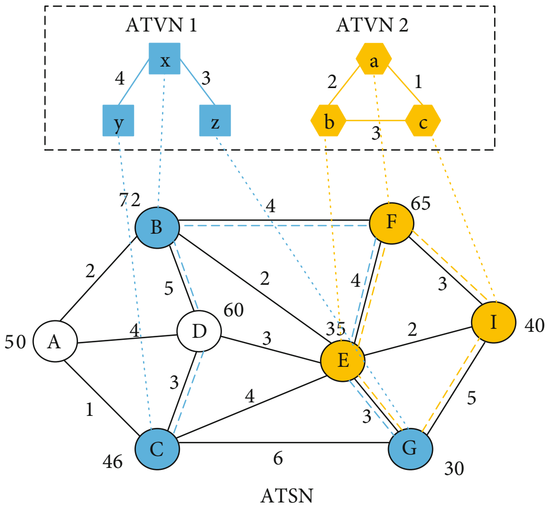


FIGURE 2: ATVN embedding.

3.2.2. Mapping Constraints and Objective Function. As shown in Figure 1, the embedding process includes node mapping and link mapping.

The constraints of node mapping are listed in Equations (1)-(4). To represent if VNode v_t is mapped onto SNode n, a Boolean variable $x(v_t, n)$ is given by

$$\forall v_t \in N_t^V, \forall n \in N^S, x(v_t, n) = \begin{cases} 1, & iff\ n = M^N(v_t) \\ 0, & \text{otherwise} \end{cases}. \tag{1}$$

To ensure that n can only map at most one v_t of the same ATVN, Equation (2) is given by

$$\sum_{n \in N^S} x(v_t, n) \le 1. \tag{2}$$

To ensure that the distance $dist(v_t, n)$ between v_t and n must be within the embedding radius φ^V of v_t, Equation (3) is given by

$$x(v_t, n) \times dist(v_t, n) \le \varphi^V. \tag{3}$$

To ensure that VNode v_t must require less resource than the resource capacity of SNode n if $x(v_t, n) = 1$, it is given by

$$VNoC(v_t) \times x(v_t, n) \leq SNoC(n). \quad (4)$$

In Equation (4), $VNoC(v_t)$ represents the practical node capacity requirement of v_t if $x(v_t, n) = 1$ and $SNoC(n)$ represent the existing node capacity of n.

The constraints of link mapping are listed in Equations (5)-(7). To represent if a SLink l^S belongs to the substrate $M^L(l_t^V)$ that maps VLink l_t^V, a Boolean variable $x(l_t^V, l^S)$ is given by

$$\forall l_t^V \in L^V, \forall l^S \in L^S, x\left(l_t^V, l^S\right) = \begin{cases} 1, & iff\, l^S \in M^L\left(l_t^V\right) \\ 0, & \text{otherwise} \end{cases}. \quad (5)$$

To ensure that l^S must have more resource capacity than what VLink l_t^V require if $x(l_t^V, l^S) = 1$,

$$\sum_{l_t^V \in L_t^V} VLiC\left(l_t^V\right) \times x\left(l_t^V, l^S\right) \leq SLiC\left(l^S\right). \quad (6)$$

In Equation (6), $VLiC(l_t^V)$ represents the practical link capacity requirement of l_t^V. $SLiC(l^S)$ represents the existing link capacity of l^S.

Lastly, we have constraints relating to flow conservation; i.e., traffic emitted by a VNode must arrive at its destination VNode. It is given by

$$\sum_{l_{ij}^S \in L^S} x\left(l_{um,t}^V, l_{ij}^S\right) - \sum_{l_{ji}^S \in L^S} x\left(l_{um,t}^V, l_{ji}^S\right) = \begin{cases} 1, & x(v_{u,t}, n_i) = 1 \\ -1, & x(v_{m,t}, n_i) = 1 \\ 0, & \text{otherwise} \end{cases}. \quad (7)$$

With these constraints of node mapping and link mapping, the major objective of the proposed algorithm is to maximize the number of the ATVNs that are successfully embedded onto ATSN. Thus, the objective function is formulated as given below:

$$\text{Maximize}\left\{ \lim_{T \longrightarrow \infty} \frac{\sum_{t=0}^{T} NUM_{Emb}(t)}{\sum_{t=0}^{T} NUM_{Arr}(t)} \right\}, \quad (8)$$

where $NUM_{Emb}(t)$ and $NUM_{Arr}(t)$ are the number of embedded ATVNs and arrived ATVNs at time t, respectively.

3.2.3. Dynamic ATVNS. The embedding results of initial ATVNs are given in advance. Then, to the dynamic ATVNs, the resource configuration and structure will evolve over time within the given time window. One existing ATVN may change its structure and the resource configuration under two circumstances. Firstly, the workload fluctuation may increase or decrease the workload on the VNode or the VLink. Thus, the resource requirement of the ATVN will accordingly change. Secondly, the changing battlefield situation information may prompt commanders to modify the mission applications defined by ATVNs so that the structure and resource configuration will change. Here, the resource configuration is referred to as the transmission rate requirement of VLinks. The structure is referred to as the topology of the ATVN. In this paper, the dynamic nature of ATVNs can be defined as node/link addition, node/link removal, node motion, and requirement change.

In Figure 3, an example is presented to show the dynamic process. There are various changes in the resource requirement and structure of ATVN 1 and 2 within the given time window $\{t \mid t \in [0, 12]\}$. In Figures 3(a) and 3(e), the initial ATVN 1 and 2 are shown with corresponding resource requirement and structure. At time $t = 1$, the resource configuration and structure of both ATVN 1 and ATVN 2 remain unchanged. However, in Figure 3(b), ATVN 1 adds one new VLink (y, z) with 6 units of transmission rate requirement at time $t = 2$. Next, in Figure 3(c), ATVN 1 adds one new VNode w with corresponding VLinks (w, x) and (w, z). These two VLinks have 2 units and 1 units of transmission rate requirement, respectively, and ensure that ATVN is an undirected graph without isolated VNodes. Similarly, dynamic nature can also include the removal of VNodes or VLinks. For example, in Figures 3(d) and 3(g), ATVN 1 removes one VLink (w, z) and one VNode z, respectively. Removal of VNode z also includes the removal of its neighboring VLinks (y, z) and (w, z). In addition, the change in the structure of ATVN can refer to as the motion of VNode. For example, in Figure 3(h), VNode c of ATVN 2 moves to another location. The dynamic nature of the ATVN also refers to as the change in the resource requirement of the VLinks. In Figure 3(f), to ATVN 2, the transmission rate requirement of VLink (a, b) decreases from 2 units to 1 units.

As shown in Figure 3, there is resource relationship among dynamic changes of ATVN 1 and ATVN 2 due to the fact that ATVNs are embedded onto the same ATSN. For example, the removal of VLink (x, z) in Figure 3(d) can satisfy the resource requirement of the addition of VLink (y, z) in Figure 3(b) or the node motion in Figure 3(h). As shown in Figure 2, VLink (x, z) is mapped onto the path $\{(B, F), (F, E), (E, G)\}$. As VLink (x, z) is removed, the occupied resource on the path will be released. Thus, the newly added VLink (y, z) can be mapped onto $\{(C, E), (E, G)\}$ if there is not enough resource on $\{(C, G)\}$. In addition, the released resource can offer some resource for the node motion if VNode c is remapped onto SNode G. In this paper, we focus on the resource relationship among dynamic changes. The proper processing of resource relationship can improve the embedding profits in the long term.

3.3. Matching Matrix. As shown in Figure 3, there are different dynamic changes of ATVNs within the given time window. Matching matrix is used to adopt the resource relationship among dynamic changes. To explain the resource relationship, we first divide dynamic changes within the given time window into two categories that are positive changes (pCs) C_t^+ and negative changes (nCs) C_t^-. And $|C_t^+|$ and $|C_t^-|$ represent the

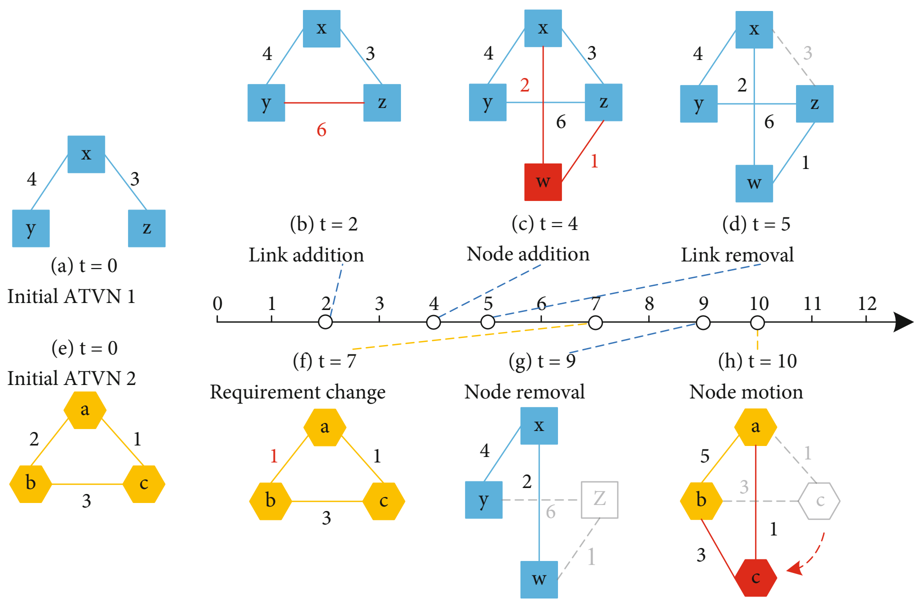


FIGURE 3: Example of dynamic changes of ATVNs.

number of C_t^+ and C_t^-, respectively. To nCs, they require more network resource to maintain the normal running of dynamic ATVNs and include link/node addition, node motion, and requirement increase. Similarly, to pCs, they release some or all of the network resource occupied by the mapped VNodes or VLinks and include link/node removal, node motion, and requirement decrease. Next, there is resource relationship between one nC and one pC. For example, one nC (requiring resource) can search for one pC (releasing resource) with closer location and higher transmission rate among pCs in order to achieve relevant resource for mapping or remapping VNodes and VLinks. In this paper, we focus on satisfying the resource requirement of nCs by matching the proper pCs to release relevant resource.

Matching matrix is represented by the notation $M(t) = [m_{ij}(t)]$ with the ith row and the jth column at time t. The I number of rows refers to as the negative VLinks (nVs) related to nCs. In addition, the J number of columns refers to as the positive VLinks (pVs) related to pCs. Hence, the dimension of the matching matrix would depend on the number of nVs and pVs at the end time t of time window. Transmission rate request and relative location are taken into account to construct the matching matrix $M(t)$. The value $m_{ij}(t)$ is calculated by the following equation, which indicates the resource relationship between one nV and one pV.

$$m_{ij}(t) = \frac{\eta_1 \cdot \exp\left(\left(TR\left(l_{t,j}^V\right) - TR\left(l_{t,i}^V\right)\right)/\eta_2\right)}{Dist\left(l_{t,j}^V, l_{t,i}^V\right)}. \quad (9)$$

In Equation (9), η_1 and η_2 are constants that coordinate the relationship between transmission rate request and relative distance. $l_{t,i}^V$ is the ith nV in the VLink group related to nCs at time t. Also, $l_{t,j}^V$ is the jth pV in the VLink group related to pCs at time t. exp $((TR(l_{t,j}^V) - TR(l_{t,i}^V))/\eta_2)$ represents that $m_{ij}(t)$ will be smaller if $l_{t,i}^V$ requires more transmission rate than $l_{t,j}^V$ releases. In this paper, $Dist(l_{t,i}^V, l_{t,j}^V)$ are defined by the shortest distance between the end VNodes of $l_{t,i}^V$ and the end VNodes of $l_{t,j}^V$. Also, $m_{ij}(t) = \infty$ if $Dist(l_{t,i}^V, l_{t,j}^V) = 0$.

One nC expects to have the matching with one pC that has enough released network resource. To evaluate if one nC has the proper matching, a new feature called *local matching value* (LMV) is calculated to quantify the suitability of a matching for a particular nC. The local matching value of the kth nC is calculated as follows:

$$\alpha_k^j(t) = \sigma_k^j(t) \sum_{i=1}^{h} \lambda_{ij}(t) m_{ij}(t), \quad (10)$$

where h is the total number of nVs related to the kth nC. $\sigma_k(t)$ is the Boolean variable that represents if the h number of nVs can make their resource requirements satisfied by the VLinks related to pCs and thus are successfully mapped or remapped. Also, $\lambda_{ij}(t)$ is the Boolean variable that represents if $m_{ij}(t)$ is not equal to ∞. Mathematically,

$$\sigma_k^j(t) = \begin{cases} 1, & \text{if } h \text{ number of nVs are satisfied} \\ 0, & \text{Otherwise} \end{cases}, \quad (11)$$

$$\lambda_{ij}(t) = \begin{cases} 1, & \text{if } m_{ij}(t) \neq \infty \\ 0, & \text{Otherwise} \end{cases}. \quad (12)$$

There can be different matching schemes for nCs and pCs. It is important to select the most proper matching scheme. However, LMV only considers one nC. Therefore, we define *global matching value* (GMV) to evaluate if all of the nCs have the proper matching with pCs. As the total number of nCs of different ATVNs is $|C_t^-|$, we calculate GMV to include the LMV of all the nCs. Mathematically,

$$\beta(t) = \sum_{k=1}^{|C_t^-|} \sum_{j=1}^{J} \alpha_k^j(t). \tag{13}$$

There is resource capacity that is defined in [21]. Like many definitions in VNE [47], both resource capacity and GMV adopt the greedy approach to select the requested object value. However, they are mostly different. First, the elements in the definitions of GMV and resource capacity are different. Resource capacity contains node capacity and link bandwidth, while GMV contains LMV in Equation (10) that only has the matching value between nVs of nCs and pVs of pCs. Second, resource capacity and GMV work at different stages of VNE. Resource capacity aims to map virtual nodes before VNs are embedded, while GMV is to deal with the dynamic changes of VNs after VNs are embedded. Finally, they have different objects. To resource capacity, virtual node with the highest resource capacity can be mapped onto the substrate node with the highest resource capacity. Then, virtual nodes can be mapped onto proper susbstrate nodes. In contrast, to GMV, as GMV is bigger, the matching between nCs and pCs are better. Thus, the proper matching can be found.

3.4. Evaluation Metrics. After successfully embedding one ATVN, it is essential to define proper metrics so as to evaluate the performance of selected embedding algorithm. Therefore, some general evaluation metrics in [47, 48] are requested to formulated, including average acceptance ratio and long-term average revenue to cost ratio.

The average acceptance ratio is an important metric that can evaluate the efficiency of successfully accepting ATVNs. Mathematically,

$$\eta_{suc}(t) = \lim_{T \longrightarrow \infty} \frac{\sum_{t=0}^{T} NUM_{Emb}(t)}{\sum_{t=0}^{T} NUM_{Arr}(t)}, \tag{14}$$

where $NUM_{Emb}(t)$ and $NUM_{Arr}(t)$ are the number of embedded ATVNs and arrived ATVNs at time t, respectively. Similarly, another metric, called average dynamic request acceptance ratio, refers to as the ratio of the number of successfully accepted dynamic requests to the total number of dynamic requests.

The long-term average revenue to cost ratio is also an important metric that can evaluate the ability of VNE in resource utilization. Mathematically,

$$R/C = \lim_{T \longrightarrow \infty} \frac{\sum_{t=0}^{T} \sum_{G_t^V \in VN_t^{Emb}} R(G_t^V)}{\sum_{t=0}^{T} \sum_{G_t^V \in VN_t^{Emb}} C(G_t^V)}, \tag{15}$$

where VN_t^{Emb} is the embedded ATVNs at time t. The revenue $R(G_t^V)$ represents the total network resource requirement (node power and link bandwidth resource) of embedded ATVNs at time t. Under the total network resource requirement, we calculate the amount of consumed network resource represented by the cost $C(G_t^V)$. Mathematically,

$$C(G_t^V) = \gamma_C \sum_{v_t \in N_t^V} tp(M^N(v_t)) + \sum_{l_t^V \in L_t^V} \sum_{l^S \in M^L(l_t^V)} b(l^S), \tag{16}$$

$$R(G_t^V) = \gamma_R \sum_{v_t \in N_t^V} tp(M^N(v_t)) + \sum_{l_t^V \in L_t^V} \sum_{l^S \in M^L(l_t^V)} b(l^S) / \left| M^L(l_t^V) \right|, \tag{17}$$

where γ_C and γ_R are weighting coefficients that coordinate the relationship between node power and link bandwidth. $tp(M^N(v_t))$ is the power resource allocated by $M^N(v_t)$ to map v_t. l_t^V is mapped onto the path $M^L(l_t^V)$ with total link number $|M^L(l_t^V)|$. A SLink l^S of $M^L(l_t^V)$ will allocate bandwidth resource $b(l^S)$ to meet the transmission rate requirement of l_t^V.

4. Heuristic Solution: DWVNE-RW

Due to the intractability of dynamic VNE problem, a heuristic relationship matrix-based dynamic wireless VNE algorithm (DWVNE-RM) is proposed to find feasible and reliable solutions in reasonable time for ATNV. DVNE-RM considers the dynamic nature of multiple ATVNs within the given time window. The dynamic nature is referred to as the changes in the structure and resource configuration of the VNodes and VLinks. The ATVNs are embedded in two stages. In the first stage, it deals with the initial reliable placement of the VNodes and VLinks (Algorithm 1). In addition, the second stage processes the dynamic placement of the VNodes and VLinks of multiple ATVNs with dynamic changes (Algorithm 2).

As the hypervisor in Figure 1 receives the first ATVN request, it will check if there are dynamic changes of ATVNs within the current given time window. The hypervisor also records the corresponding information (e.g., type, time) of changes. After the current time window ends, it will repeat the above action within the next time window. Upon arrival of each ATVN request, the hypervisor will invoke the Algorithm 1 with resource requirement as input. Algorithm 1 also requires the resource availability at each SNode and SLink, which can be achieved by the global view of SDN based framework in Figure 1. As the time window ends, the hypervisor will invoke Algorithm 2 with the current embedding solutions and the updated resource availability of ATSN as the input. All the information of dynamic changes is also forwarded to Algorithm 2. Based on such information, Algorithm 2 remaps the existing VNodes and VLinks or maps the new ones with the objective to maximize the local matching value (LMV) and global matching value (GMV). Following Algorithm 2, the resource availability of the SNodes and SLinks is updated. However, the current embedding solution remains unchanged if no changes are made in the ATVNs. The subsections discuss the details of Algorithm 1 and Algorithm 2.

```
Data: Substrate network:$G^S = (N^S, L^S)$, Dynamic ATVN $G_t^V = (N_t^V, L_t^V)$;
1. $M \longleftarrow \varnothing; V \longleftarrow N_t^V$
2. while $V \neq \varnothing$ do
3.     Calculate $NoV(v_t)$ by Equation (18);
4.     Select VNode $v_{u,t}$ with the max $NoV(v_{u,t})$;
5.     Calculate $NoV(n)$ by Equation (13), (14) and (19);
6.     Select the SNode $n$ with the max $NoV(n)$;
7.     Map $v_{u,t}$ onto $n$;
8.     If not mapped, reject the $G_t^V$;
9.       $M \longleftarrow M \cup \{v_{u,t}\}; V \longleftarrow V - \{v_{u,t}\}; Li \longleftarrow \varnothing$;
10.     foreach VNode $v_{m,t} \in M$
11.         $Li \longleftarrow Li \cup \{l_{um,t}^V\}$ when $l_{um,t}^V \in L_t^V$;
12.         while $Li \neq \varnothing$ do
13.             Select $l_t^V \in Li$ with the max $TR(l_t^V)$;
14.             $Li \longleftarrow Li - \{l_t^V\}$;
15.             Ignore some SLinks by Equation (16);
16.             Map $l_t^V$ by the approach in [27];
17.             If not mapped, reject the $G_t^V$;
18.     end
19. end
20. Update the resource availability of $G^S$;
```

Algorithm 1: Initial Reliable Wireless Virtual Network Embedding Algorithm.

```
Data: Substrate network:$G^S = (N^S, L^S)$, Dynamic changes $c_t$;
1. Sort dynamic changes into $C_t^+$ and $C_t^-$;
2. foreach dynamic change $c_t$
3.     If $c_t$ is link/node removal or addition, turn it into related nVs or pVs;
4.     If $c_t$ is requirement decrease or increase, turn it into one nV or pV;
5.     If $c_t$ is node motion, turn it into pVs or nVs related to the original VNode and the new one, respectively;
6. Record the number $I$ and $J$ of nVs and pVs;
7. Construct matching matrix $M(t)$ by Equation (9);
8. Match the nVs with the pVs in a greedy way;
9. Remap all the nCs with the greedy matching by Algorithm 1;
10. Calculate LMV $\alpha_k^j(t)$ of $c_t$ and GMV $\beta(t)$;
11. $StopCond = FALSE$;
12. while $StopCond = FALSE$ do
13.         $StopCond = TRUE$;
14.         foreach nV $l_{t,i}^V$
15.             Match $l_{t,i}^V$ with another pV $l_{t,j}^V$; Release the original pV;
16.             If the pV $l_{t,j}^V$ has been matched with its original nV, rematch its original nV with the released pV;
17.             Remap the nV $l_{t,i}^V$; Calculate LMV $\alpha_k^j(t)$ of $c_t$ and GMV $cb$;
18.             if $cb < \beta(t)$
19.                 Undo the last matching in Line 15;
20.             else
21.                 Consider current matching as valid;
22.                 $StopCond = FALSE$; $\beta(t) = cb$;
23. end while
24. Update the resource availability of $G^S$;
```

Algorithm 2: Relationship matrix-based dynamic wireless VNE algorithm.

4.1. Initial Reliable Wireless VNE. In this section, initial reliable wireless VNE is completed as presented in Algorithm 1. Before embedding, the hypervisor receives the total number of time slots required to run the ATVN request and the given time window, respectively. Due to the unique features (i.e., vulnerable wireless channels, broadcast property, and malicious attacks), it is essential to fully considerate the complex interference when mapping VNodes and VLinks. Thus, Algorithm 1 consists of the following three parts:

(1) Node mapping against interference (Lines 3-8). Node mapping requires to find proper SNodes to map VNodes of ATVNs. Thus, reliable node ranking approach, incorporating interference information, can improve the reliability. The ranking values of VNode and SNode are represented as follows, respectively:

$$NoV(v_t) = \sum_{l_t^V \in L_{v_t}} TR\left(l_t^V\right), \tag{18}$$

$$NoV(n) = \sum_{l^S \in L_n} b\left(l^S\right) \log_2 \left(1 + \frac{tp(n) g\left(l^S\right)}{Inf(n) + Inf\left(l^S\right)}\right). \tag{19}$$

Equation (18) represents the total transmission rates in the set, denoted as L_{v_t}, of neighboring VLinks of VNode v_t. In Equation (19), we take the complex interference into consideration by calculating the max transmission rate of one SLink l^S according to the famous *Shannon formula*. $Inf(n)$ and $Inf(l^S)$ represent the practical interference power of SNode n and SLink l^S, respectively. Thus, Equation (19) represents the total transmission rates in the set, denoted as L_n, of neighboring SLinks of SNode n. As given in Lines 3 to 4, when the node mapping process of one ATVN request starts, it will calculate the ranking values according to Equation (18) and (19). Then, as given in Lines 5 to 7, the VNode, with the max ranking value, will tend to be mapped onto the SNode, with the max ranking value, under the constraints of Equation (13) and (14).

(2) Link mapping against interference (Lines 11-17). In the wired VNE algorithms, it tends to adopt the shortest-path or multi-commodity flow approach to map VLinks. However, the unstable link quality of wireless SLinks makes these approaches not appropriate for ATNs considering the complex interference. Thus, it adopts a link mapping based on opportunistic routing and differentiated transmission rates in our previous work [27]. To decrease the time complexity, the SLinks that fail to meet the constraint Equation (16) will not be considered for mapping VLinks in the preprocessing process of ATSN as given in Line 15

(3) One-stage mapping (Lines 1-20). The relationship between node mapping and link mapping is two-stage mapping or one-stage mapping. Two-stage mapping processes the node mapping and link mapping in two ordered and isolated stages, which cannot guarantee the optimal or suboptimal embedding assignment per VN in many cases [49]. In this paper, we propose a one-stage mapping strategy that can adopt the correlation between node mapping and link mapping to improve the resource utilization in the long run

Theorem 1. *The time complexity of Algorithm 1, with* $|N_t^V|$ *number of VNodes,* $|L_t^V|$ *number of VNodes,* $|N^S|$ *number of SNodes, and* $|L^S|$ *number of SLinks at any time t, is* $O(|N_t^V|(|N^S|^2 + |L_t^V|(|N^S| \log(|N^S|) + |L^S|)))$.

Proof. Upon arrival of each ATVN at the hypervisor, Algorithm 1 is invoked for the initial mapping of the VNodes and VLinks. First, calculation of node ranking values of VNodes is given in Algorithm 1 in Line 3, which requires at most a running time of $O(|N_t^V|(|N_t^V| - 1)) = O(|N_t^V|^2)$ since any VNode has at most $|N_t^V| - 1$ number of VLinks. The selection in Line 4 has a running time of $O(|N_t^V|)$. Similarly, the running time of calculation of node ranking values of SNodes is at most $O(|N^S|(|N^S| - 1)) = O(|N^S|^2)$. The selection in Line 6 has a running time of $O(|N^S|)$. Next, Line 11 has at most a running time of $O(|N_t^V|(|N_t^V| - 1)/2) = O(|N_t^V|^2)$ since there are at most $|N_t^V|(|N_t^V| - 1)/2$ number of VLinks in the set L_t^V. Also, the running time of the inner loop (Lines 12 to 18) is determined by Line 16 that has a running time of $O(|N^S| \log(|N^S|) + |L^S|)$ in [27]. Thus, it has a total running time of $O(|L_t^V|(|N^S| \log(|N^S|) + |L^S|))$. Therefore, the total running time of Algorithm 1 is concluded as follows:

$$O\left(|N_t^V|\left(|N^S|^2 + |N_t^V|^2 + |N_t^V|^2 + |L_t^V|\left(|N^S| \log(|N^S|) + |L^S|\right)\right)\right). \tag{20}$$

Due to the fact that $|N_t^V|$ is much less than $|N^S|^2$, the time complexity at time t can be rewritten as $O(|N_t^V|(|N^S|^2 + |L_t^V|(|N^S| \log(|N^S|) + |L^S|)))$. □

4.2. Dynamic Embedding of ATVNs. Following the initial reliable mapping of VNodes and VLinks, Algorithm 2 is proposed to map or remap the corresponding VNodes and VLinks of dynamic ATVNs in order to improve the acceptance ratio.

Algorithm 2 firstly deals with the dynamic changes as given in Lines 1-9. Then, Line 10 represents the calculation of the LMV of one nC and GMV of all nCs, respectively, to execute Algorithm 2. As given in Line 11, *StopCond* is defined as a conditional variable that indicates if the algorithm execution is continued or not. In the beginning stage of each iteration, *StopCond* is set to be *TRUE* assuming that the iteration will not be continued because further maximization of β cannot be found, as given in Line 13. However, in Line 22, if the further maximization is possible due to at least one rematching between nV and another pV, *StopCond* variable will be reset to be *FALSE*.

As given in Lines 1-6, the hypervisor runs Algorithm 2 and deals with the dynamic changes collected within the given time window. It firstly sorts these changes into two categories (pCs C^+ and nCs C^-) as given in Line 1. C^+ includes link/node removal, requirement decrease, and node motion. Similarly, C^- includes link/node addition, requirement increase, and node motion. Then, they are transformed into corresponding VLinks (nVs and pVs) in Lines 2-6. In Line 3, the removal or the addition of VLink can be transformed into one VLink, called pV or nV, with its transmission rate request and specific location of two end nodes. Also, node removal and node addition can be transformed into neighboring VLinks with corresponding information of link request and node location. In addition, in Line 4, both the requirement decrease and the requirement increase can be transformed into one VLink with corresponding information. Next, in Line 5, node motion is considered the combination of node removal and node addition. Node removal is related to the original VNode without movement, while node addition is related to the new VNode after the movement.

To get the proper matching between nCs and pCs, we first make a matching according to the matching matrix $M(t)$ in a greedy way as presented in Lines 7-8. Based on the greedy matching, the VNodes and VLinks related to the nCs are mapped or remapped by adopting Algorithm 1, as given in Line 9. Then, the original LMV and GFV are calculated in Line 10 according to Equation (10) and Equation (13). Next, in each while loop as given in Line 12, each nV is examined with all pV as follows. At first, it is assumed that the nV $l_{t,i}^V$ is rematched with another pV $l_{t,j}^V$ as given in Line 15. If any other nV is already matched with this pV $l_{t,j}^V$, it will be rematched with the original pV that is matched with the nV $l_{t,i}^V$ as given in Line 16. Following the new rematching, the LMV and GMV of the nVs and pVs are calculated in Line 17. If the previous GMV value $\beta(t)$ is less than the newly calculated GMV value cb, the current rematching is considered a valid and better remapping solution of dynamic changes. However, if the previous GMV value $\beta(t)$ is greater than the newly calculated GMV cb, the current rematching of nV is discarded or cancelled. For the sake of better understanding, the proposed DWVNE-RM algorithm is explained with an example in Section 4.3.

Theorem 2. *The time complexity of Algorithm 2, with I number of nCs, J number of pCs, is* $O(I \log_2(I) + J(|N^S|^2 + J(|N^S| \log_2(|N^S|) + |L^S|)))$.

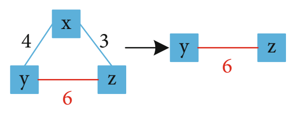
Proof. Algorithm 2 is executed by transforming nCs and pCs into nVs and pVs as given in Lines 1-6, which requires the running time of $O(J+I)$. Following this, the running time of Lines 7-11 is mainly determined by Lines 8-9. As given in Line 8, the matching between nCs and pCs has the time complexity of $O(J + I \log_2(I))$. Also, the time complexity of remapping is at most $O(J(|N^S|^2 + J(|N^S| \log(|N^S|) + |L^S|)))$. Next, the time complexity of Lines 12-23 is determined by the inner *foreach* loop as given in Lines 14-17. The *foreach* loop has the running time of $O(J(|N^S|^2 + (|N^S| \log(|N^S|) + |L^S|)))$ that is determined by Line 17. In conclusion, Algorithm 2 has the total time complexity of $O(I \log_2(I) + J(|N^S|^2 + J(|N^S| \log(|N^S|) + |L^S|)))$. □

Theorem 3. *Maximizing the GMV maximizes the average acceptance ratio.*

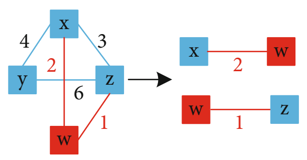
Proof. As represented in Equation (14), average acceptance ratio refers to as the ratio of the number of accepted ATVN requests to the total number of ATVN request received by the hypervisor at any time t. Here, GMV refers to as the proper processing of dynamic changes in the structure and the resource configuration of ATVN requests. Dynamic changes will create new demands of mapping or remapping VNodes and VLinks of corresponding ATVNs. If the new demands are not properly processed, the corresponding ATVNs will be denied, and thus the acceptance ratio will decrease. For example, the hypervisor collects the dynamic information that one ATVN has an nC that adds a VLink (nV) with 6 units of resource requirement and another ATVN has a pC that removes a VLink (pV) with 8 units of resource requirement. There is resource relationship between the nV and pV. Thus, it can match the nV with the pV so that the nC can be accepted by adopting Algorithm 2. Then, the ATVN with the nV will be accepted again. However, considering the limited network resource, the nV may not be accepted if the resource, released by the pV, is used by other ATVNs but not enough to embed these ATVNs. Inspired from such scenario, we are using matching matrix to match nVs with proper pVs to accept more dynamic ATVNs. In other words, we are properly processing the dynamic changes, which are represented by the maximization of GMV and thus maximizes the average acceptance ratio. □

Theorem 4. *Algorithm 2 gives near optimal result for the dynamic ATVN due to optimal resource allocation of changing VNodes and VLinks.*

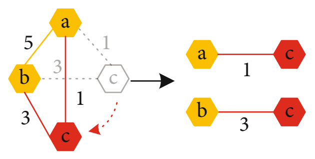
Proof. The objective of Algorithm 2 is to properly process the resource relationship among dynamic changes (nCs and pCs) in order to maximize the average acceptance ratio. It follows the greedy approach for mapping or remapping VNodes and VLinks of nCs by exploiting the related resource of pCs. For each nC, Algorithm 2 transforms it into several nVs by considering the resource requirement of VLinks and the location of VNodes. Then, to the nVs, Algorithm 2 selects the optimal pVs from pCs that release related resource to map or remap VNodes and VLinks, by using the matching matrix where the location and requirement of resource are considered. The same procedure is followed to find the next best matching between the nVs of nCs and related pVs of pCs. Further, Algorithm 2 also makes original reliable and optimal mapping of VNodes and VLinks as given in Algorithm 1. With the intention of processing the resource relationship among dynamic changes, the resource availability of pCs is properly matched with the related resource requirement of nCs. The intention of making locally optimal matching of nVs for each nC is to find the global optimal matching for all the dynamic changes. In doing so, Algorithm 2 can produce a near-optimal matching in this greedy way. □



(a) nC1 link addition

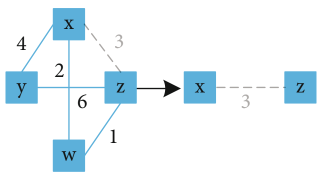


(b) nC2 node addition

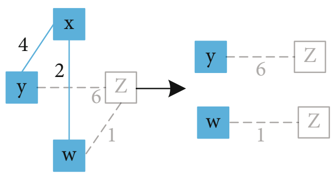


(c) nC3 node motion

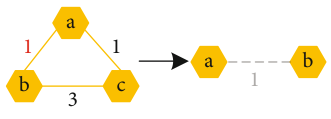
Figure 4: Transforming negative changes.



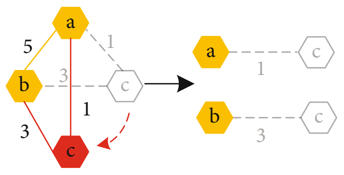
(a) pC1 link removal



(b) pC2 node removal



(c) pC3 requirement change



(d) pC4 node motion

Figure 5: Transforming positive changes.

4.3. Example of DWVNE-RM. For the sake of better understanding, let us take an example as given in Figures 4–7. As shown in Figure 3, the hypervisor records 4 number of nCs within the given time window. Figure 4 represents the process of transforming these nCs into nVs. First, as shown in Figure 4(a), the nC called link addition happens at ATVN 1 with 3 VNodes (x, y, and z) and 2 VLinks. Link addition can be transformed into one nV with 6 units of resource requirement and corresponding VNodes (y and z). Second, as shown in Figure 4(b), ATVN 1 has another nC that is node addition at time $t = 4$. The node addition can be transformed into two nVs. One nV has 2 units of resource requirement and corresponding VNodes (x and w). Another nV has 1 unit and VNodes (w and z). In addition, as shown in Figure 4(c), ATVN 2 has another nC caused by node motion. Thus, VNode c moves to a new location, which can be transformed into two nVs. One nV has 1 unit and VNodes (a and c). Another nV has 3 units and VNodes (b and c).

As shown in Figure 3, the hypervisor also records 3 number of pCs within the given time window. These pCs are represented in Figure 5. In Figure 5(a), ATVN 1 has a pC called link removal that can be transformed into a pV with 3 units and VNodes (x and z). In Figure 5(b), ATVN 1 has a pC called node removal. The change can be transformed into two pVs. One pV has 6 units and VNodes (y and z). Another pV has 1 unit and VNodes (w and z). Also, in Figure 5(c), ATVN 2 has an nC called the requirement decrease of VLink. Then, it is transformed into one nV with 1unit and VNodes (a and b). Next, in Figure 5(d), ATVN 2 has a pC caused by node motion. VNode c moves from the original location, which can be transformed into two pVs. One has 1unit and VNodes (a and c). Another has 3 units and VNodes (b and c).

Figures 6(a) and 6(b) list out all of the nVs and pVs with corresponding resource information. Based on Figures 6(a) and 6(b), we can get the matching matrix by using Equation (9), where η_1 and η_2 are set to be 10 and 4, respectively, as given in Figure 6(c). Meantime, in this paper, the node location is defined by x and y coordinates. Thus, the locations of VNodes (x, y, z, and w) in ATVN1 are assumed to be (3, 1), (2, 5), (7, 2), and (8, 6), respectively. Also, in ATVN 2, the locations of VNodes (a and b) are (5, 2) and (3, 4), respectively. And the original location of VNode c is (6, 8), while the new location is (8, 4). For example, the matching value between nV1 and pV1 is about 1.1 according to Equation (9). The distance between nV1, with nodes y and z, and pV1, with nodes x and z, is defined by the shortest distance between their end nodes, which is $\sqrt{(3-2)^2 + (5-1)^2} + 0 \approx 4.12$. Then, the calculation of matching value is $10 \exp((3-6)/4)/4.12 \approx 1.1$. Also, if the matching value is ∞, it means the nV and the matched pV are originally the same one.

nCs	nVs	Resource	VNodes
nC1	nV1	6	(y, z)
nC2	nV2	2	(x, w)
	nV3	1	(w, z)
nC3	nV4	1	(a, c)
	nV5	3	(b, c)

(a) nVs with resource demands

pCs	pVs	Resource	VNodes
pC1	pV1	3	(x, z)
pC2	pV2	1	(a, b)
pC3	pV3	6	(y, z)
	pV4	1	(w, z)
pC4	pV5	1	(a, c)
	pV6	3	(b, c)

(b) pVs with resource demands

		pV1	pV2	pV3	pV4	pV5	pV6
nC1	nV1	1.1	0.8	∞	0.5	0.4	0.6
nC2	nV2	3.1	1.0	3.3	1.9	1.5	2.2
	nV3	2.3	1.4	5.7	∞	2.1	2.3
nC3	nV4	3.7	2.0	5.4	2.5	2.2	2.3
	nV5	1.9	1.7	5.8	0.9	0.8	2.2

(c) Matching matrix

Figure 6: Matching negative VLinks and positive VLinks.

		Greedy	Iteration 1	Iteration 2
nC1	nV1	pV3	pV3	pV3
nC2	nV2	pV1	pV6	pV2
	nV3	pV4	pV4	pV4
nC3	nV4	pV6	pV1	pV1
	nV5	pV2	pV2	pV6
		$\beta = 7.1$	$\beta = 7.6$	$\beta = 6.9$

Figure 7: Results of iterations.

Figure 7 shows the results of iterations. In the column called greedy, Algorithm 2 match nVs with pVs in a greedy way after the calculation of matching matrix as given in Figure 6(c). For instance, pV3 is selected for nC1 due to the fact that their matching value is the biggest in the first row of matching matrix. Also, the matching will not be changed during the iterations because their matching value is ∞. For the rest of nVs, it will match them with the pVs that have the biggest matching values. Then, Algorithm 1 will remap the VNodes and VLinks of nCs based on the greedy matching. And it is assumed that all the nCs can be well processed. According to Equation (10), the LMV of nC1 is calculated by $1 \times 0 \times \infty = 0$. Similarly, LMVs of nC2 and nC3 are 3.1 and 4, respectively. Then, the GMV β of the dynamic VNs is calculated by $0 + 3.1 + 4 = 7.1$, using Equation (13). The objective of Algorithm 2 is to maximize the value of β. For this, each nV is examined, which can have some iterations. For example, nV1 is rematched with pV6 in the column called iteration 1. The new LMV of nC2 is calculated as $1 \times 1 \times 2.2 \times 1 \times 0 \times \infty = 2.2$. Considering the LMV of each nC, the new GMV β value is calculated as 7.6. The new β value is bigger than the previous β value of 7.1. Hence, the current matching solution is considered a efficient one. In addition, in the column called iteration 2, the same procedure of rematching nVs is followed to further maximize the β value. The new β value is 6.9 that is smaller than 7.6. Thus, the current matching solution is considered an invalid one. There can be other iterations to maximize the GMV β. If other iterations have not a bigger β, iteration 1 will be considered the final solution to process the resource relationship between nCs and pCs at current time instance.

5. Simulation Experiment

In this section, DWVNE-RM is evaluated through extensive simulations. Firstly, the simulation setup is presented in detail. Then, the compared algorithms are clearly described, followed by simulation results.

5.1. Simulation Setup. The network topology of ATSN and ATVN is generated by an improved Salam network topology random generation algorithm [18]. Specific network parameters are presented in Tables 1 and 2 for ATSN and ATVN, respectively. In Table 1, the unit of SNode power is *W*. The unit of SLink bandwidth is *MHz*. The power range is based

Table 1: ATSN parameters.

Parameter	Description
Scope	$200\,\text{km} \times 200\,\text{km}$
Number of SNodes	50
Number of SLinks	139
Available SNode power among flight formations	[50, 100], uniform distribution
SLink bandwidth	[20, 50], uniform distribution
SNode position	[0, 200], x and y coordinates

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 VNodes	[3, 5], Uniform distribution
VLink connection probability	0.5
VLink transmission rate	[3, 8], Uniform distribution
VNode position	[0, 200], x and y coordinates

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*.

As shown in Figure 1, ATSN is formed by substrate nodes that are regarded as lead aircrafts of different flight formations. Lead aircrafts are deployed within the scope $200\,km \times 200\,km$ and followed by several wingman. The altitude equals 10000 m over the sea level. The minimum of a lead aircraft is set to be 150 m/s. Considering the network dynamics of airborne tactical network, there may be two different conditions. First, one lead aircraft will sometimes change its flight trajectory to do flying movements so that the corresponding substrate node may be disconnected. In this case, the lead aircraft can turn over its mission to nearby wingman to keep connected with ATSN until it become normal. Next, one lead aircraft may change its position so that its adjacent substrate links will change in ATSN and corresponding virtual links need to be remapped. In this case, Algorithm 1 in Section 4.1 can be adopted to remap corresponding virtual links.

Other simulation settings are made for ATSN and ATVN. Each SNode (n) has enough basic power $p_b(n)$ to communicate within flight formation. Environmental noise is white Gaussian noise with power 10^{-6} W. Next, in practical wireless environment, link gain includes some parameters (e.g., antenna gains, multipath models). To simplify the analysis process, in this paper, link gain of a SLink is set 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 addition, to ATVN, φ^V is set to be 100 km. η_1 and η_2 in Equation (9) are set to be 10 and 1, respectively. Also, both γ_C and γ_R in Equation (16) and (17) are set to be 1.

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 20000 time units on the analysis software which is MATLAB R2019b.

5.2. Compared Algorithms. Table 3 lists out all the heuristic VNE algorithms with corresponding dynamic embedding strategies for experimental simulations.

5.3. Simulation Results. Simulation results are compared between DWVNE-RM and other three heuristic dynamic embedding algorithms. In addition, malicious attacks are not assumed to be intense. Thus, the intensity of general interference is denoted by $tp_b(n)/(Inf(n) + Inf(l^S) = 1$, described as 0 dB $(S/N = 10\,\lg\,(tp_b(n)/(Inf(n) + Inf(l^S)))$. The main performance metrics are average acceptance ratio, average dynamic request acceptance ratio, long-term average revenue to cost ratio, and average execution time. All the average values of corresponding metrics are obtained by simulating the algorithms for multiple times.

The evaluation of algorithms is to test their ability to handle the dynamic resource demands of ATVNs. According to the ATVN arrival rate in Table 2, there is about 1000 number of ATVNs during the running time of 20000 time units. For these running ATVNs, there is a large number of requests to modify the structure or the resource requirement of the ATVNs. The given time window is set to be 500 time units. Within the given time window, the dynamic requests are collected and processed. On one hand, the performance of different algorithms is compared (Figures 8–11), under the condition that 10% of the ATVNs have dynamic requests. On the other hand, the performance is compared (Figures 12–14) under the condition that the number of ATVNs with dynamic requests changes from 10% to 50%.

As shown in Figure 8, the performance of the proposed DWVNE-RM algorithm is compared with other three algorithms in terms of average acceptance ratio. Acceptance

Table 3: Compared algorithms.

Notation	Description
DWVNE-RM	The proposed algorithm in this paper. Dynamic embedding considers the resource relationship among dynamic changes of different VNs
DYVINE [29]	Dynamic embedding considers the dynamic changes of single VN by calculating fitness values of the VN resource demand and the resource availability of SN
DQDMA [30]	Dynamic embedding considers dynamic changes of single VN, addressing the issue of optimizing the quality of service (QoS) performance of each accepted VN
DVNMA [35]	Dynamic embedding considers dynamic changes of single VN with the objective to minimize the embedding cost

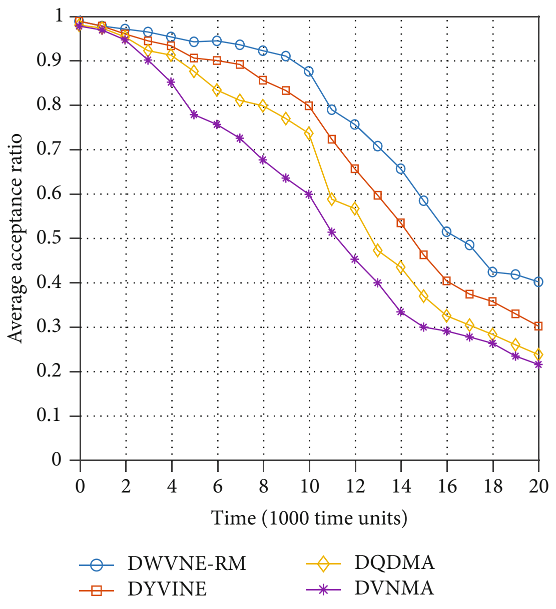


Figure 8: Average acceptance ratio.

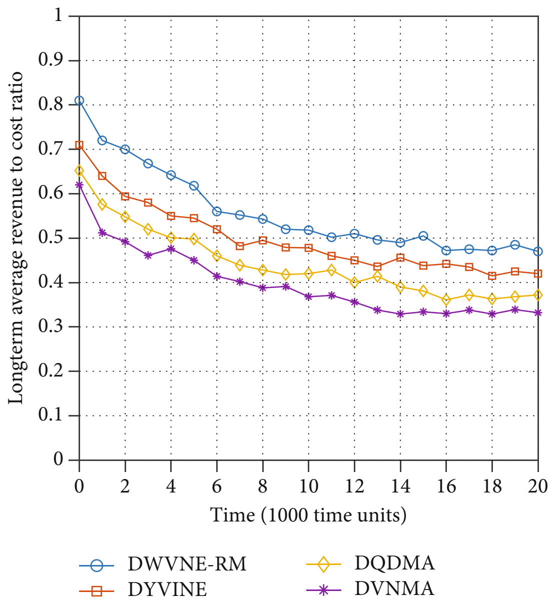


Figure 9: Long-term average revenue to cost ratio.

ratio refers to the accepted ATVNs to total ATVNs, which has direct impact on the embedding profits. In the beginning stage of simulation, the average acceptance ratio is more than 0.95 due to the adequate network resource of ATSN. However, the average acceptance ratio decreases as the simulation time goes on. This is because there are new ATVNs and some of accepted ATVNs have dynamic requests. It is observed that the average acceptance ratio of DWVNE-RM is approximately 0.48 in the final stage of simulation, whereas the average acceptance ratio is below 0.38, 0.32, and 0.30 in case of DYVINE, DQDMA, and DVNMA. The higher average acceptance ratio of DWNE-RM is due to the proper processing of the resource relationship between positive changes and negative changes in ATNs. Thus, ATVNs with dynamic changes may be more likely to be successfully re-accepted.

In order to compare the efficiency in terms of long-term average revenue to cost ratio, the data relationship regarding the network resource demand (revenue) of ATVNs and the allocated network resource (cost) of ATSN is simulated as shown in Figure 9. The long-term average revenue to cost ratio decreases as simulation time extends, which is because there are more ATVNs and less network resource. Also, it is observed that the proposed DWVNE-RM algorithm outperforms other three algorithms in terms of long-term average revenue to cost ratio. The ratio decreases from 0.81 to 0.48, when the simulation time increases from 0 to 20000 time units. The ratio in case of DYVINE, DQDMA, and DVNMA is below 0.42, 0.37, and 0.33. The major reason behind such efficiency is due to the fact that it can get more revenue with more accepted ATVNs and achieve less cost by finding shorter reliable paths. In addition, the more accepted ATVNs are due to the proper processing of relationship between positive and negative changes, which can reaccept more ATVNs with dynamic changes.

As shown in Figure 10, the performance of DWVNE-RM is compared with other three algorithms in terms of average dynamic request acceptance ratio that refers to as the ratio of the number of accepted dynamic requests to the total number of dynamic requests. It is observed that the ratio decreases as the simulation time extends. The reason of the fact is that there is less network resource to remap the VNodes and VLinks of the dynamic requests and some dynamic requests are not accepted. In case of DWVNE-RM, the average dynamic request acceptance ratio decreases from more than 0.95 to nearly 0.70, when the simulation

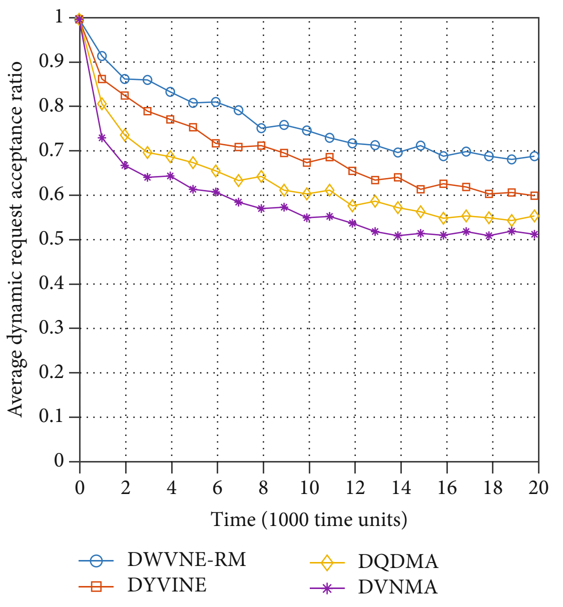


Figure 10: Average dynamic request acceptance ratio.

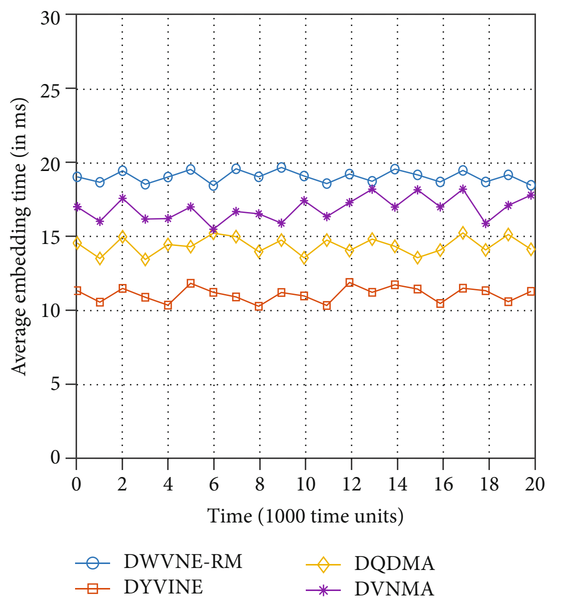


Figure 11: Average embedding time.

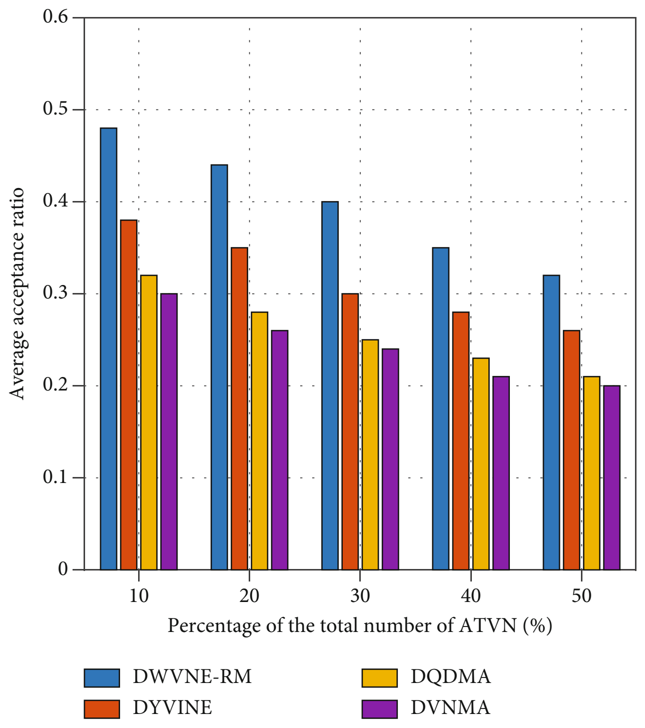


Figure 12: Effect of dynamic ATVN changes on average acceptance ratio.

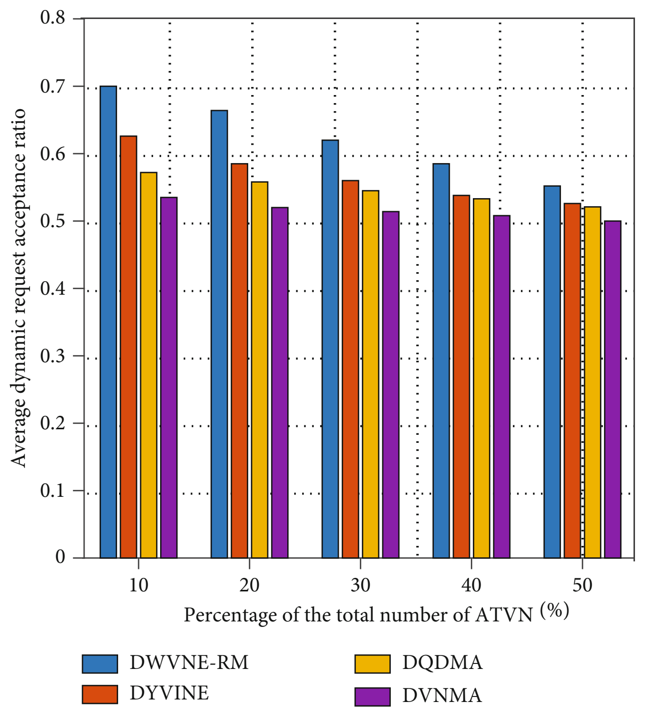


Figure 13: Effect of dynamic ATVN changes on average dynamic request acceptance ratio.

time extends from 0 to 20000 time units. In the beginning stage of simulation, the ratio of other three algorithms lies in the range of 0.80 to 0.95. However, the ratio decreases to less than 0.62, when the simulation time is 20000 time units. The higher average dynamic request acceptance ratio is due to proper processing of relationship between negative changes and positive changes, which can provide relevant network resource to map or remap more VNodes and VLinks of dynamic requests.

Figure 11 shows the comparison of all four algorithms in terms of average embedding time. In this work, the average embedding time includes the average time taken by the dynamic VNE algorithm to embed one current VN and process the corresponding dynamic requests about the changes made in the structure and configuration. In case of DWVNE-RM, the average embedding time is approximately 19.80 ms as the simulation time extends from 0 to 20000

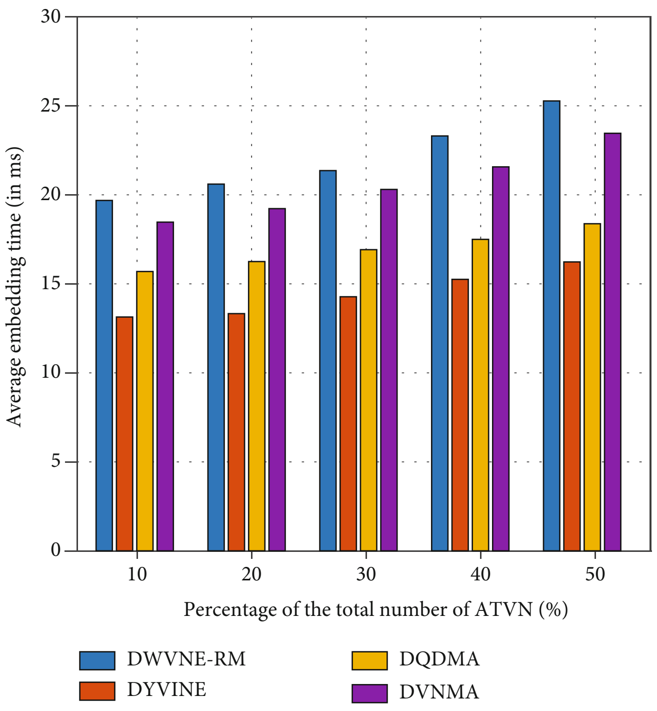


Figure 14: Effect of dynamic ATVN changes on average embedding time.

time units. However, for other three dynamic VNE algorithms, the average embedding time is approximately below 13 ms, 16 ms, and 18 ms in case of DYVINE, DQDMA, and DVNMA, respectively. The major reason behind the higher average embedding time is due to the processing of resource relationship between negative and positive changes when mapping or remapping VNodes and VLinks in ATNs.

Figure 12 shows the effect of dynamic ATVN changes on average acceptance ratio. In Figure 12, x-axis represents the percentage of the total number of ATVN during the simulation time, which refers to as the number of ATVN with dynamic changes. This observation includes the simulation result of average acceptance ratio in Figure 8 as the percentage is 10% and the simulation time is 20000 time units. It is observed that the average acceptance ratio decreases as the percentage increases, which is due to the fact that there is not enough network resource to map or remap the corresponding VNodes and VLinks of dynamic changes as the number of changes increases. In case of DWVNE-RM, the average acceptance ratio decreases from 0.48 to nearly 0.32 as the percentage increases from 10% to 50%. In addition, the average acceptance ratio of other three dynamic VNE algorithms is lower than the proposed DWVNE-RM as the percentage increases. The major reason of the higher average acceptance ratio in case of DWVNE-RM is due to the proper processing of the resource relationship between negative changes and positive changes, which can remap more VNodes and VLinks from dynamic changes and re-accept more ATVNs.

The effect of dynamic ATVN changes on average dynamic request acceptance ratio is shown in Figure 13. Similarly, the observation includes the simulation result of average dynamic request acceptance ratio in Figure 10 as the percentage is 10% and the simulation time is 20000 time units. It is observed that the ratio decreases as the percentage increases from 10% to 50%. In case of the proposed algorithm DWVNE-RM, the ratio decreases from 0.70 to nearly 0.55 when the percentage increases from 10% to 50%. For other three algorithms, the ratio is below nearly 0.53, 0.52, and 0.50 in case of DYVINE, DQDMA, and DVNMA, respectively, as the percentage is 50%. The higher ratio in case of DWVNE-RM is due to the proper processing of resource relationship, which maps or remaps more VNodes and VLinks of dynamic ATVN changes and thus accepts more dynamic requests.

Figure 14 shows the effect of dynamic ATVN changes on average embedding time. This observation also includes the simulation result of average embedding time in Figure 11 as the percentage is 10% and the simulation time is 20000 time units. It is observed that the embedding time increases as the percentage increases. This is because the algorithms need to map or remap more VNodes and VLinks as the percentage increases from 10% to 50%. In case of the proposed algorithm DWVNE-RM, the average embedding time increases from 19.80 *ms* to nearly 25.50 ms as the percentage increases, whereas the average embedding time is below 17.40 ms, 18.30 ms, and 23.30 *ms* in case of DYVINE, DQDMA, and DVNMA, respectively, as the percentage is 50%. The higher average embedding time in case of DWVNE-RM is also due to the proper processing of resource relationship between negative changes and positive changes when reliably mapping or remapping VNodes or VLinks of dynamic changes.

6. Conclusion

In this paper, we have studied the problem of embedding airborne tactical virtual networks (ATVNs) with changing structure and resource demand or configuration. A heuristic relationship matrix-based dynamic wireless virtual network embedding (DWVNE-RM) algorithm is proposed to deal with the dynamic nature of ATVNs. In this paper, the dynamic nature refers to as the change in resource configuration and network topology of ATVNs. With different resource demands on substrate network, the changes can be divided into two categories that are negative changes and positive changes. When given an ATVN request, a reliable wireless approach is adopted to complete the initial ATVN embedding. As the ATVNs change, the resource relationship between negative and positive changes is adopted to remap corresponding virtual nodes and links to maximize the acceptance ratio and resource utilization. Besides, we have evaluated the proposed algorithm by comparing with the existing approaches.

As the part of our future work, we plan to extend this work to deal with dynamic changes of substrate network. In addition, we intend to upload the DWVNE-RM algorithm in a real testbed environment and evaluate it through a prototype implementation.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

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

References

[1] B. N. Cheng, F. J. Block, B. R. Hamilton et al., "Design considerations for next-generation airborne tactical networks," *IEEE Communications Magazine*, vol. 52, no. 5, pp. 138–145, 2014.

[2] D. A. Deptula, H. R. Penney, L. A. Stutzriem, and M. Gunzinger, *Restoring America's Military Competitiveness: Mosaic Warfare*, Arlington, The Mitchell Institute for Aerospace Studies, 2019.

[3] 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 of Things Journal*, vol. 7, no. 12, pp. 11725–11734, 2020.

[4] W. Joy, 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 *MILCOM 2015-2015 IEEE Military Communications Conference*, pp. 991–996, Tampa, FL, USA, October 2015.

[5] A. Tiwari, A. Ganguli, A. Sampath et al., "Mobility aware routing for the airborne network backbone," in *MILCOM 2008-2008 IEEE Military Communications Conference*, p. 2008, San Diego, CA, USA, November 2008.

[6] T. Strayer, R. Ramanathan, D. Coffin et al., "Mission-centric content sharing across heterogeneous networks," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, Honolulu, HI, USA, February 2019.

[7] J. Miao, N. Lv, Q. Gao, K. Chen, and X. Wang, "Fault-tolerant embedding algorithm for node failure in airborne tactical network virtualization," *IEEE Access*, vol. 10, pp. 60558–60571, 2022.

[8] 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.

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

[10] N. N. Sapavath and D. B. Rawat, "Wireless virtualization architecture: wireless networking for internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5946–5953, 2020.

[11] C. Liang and F. R. Yu, "Wireless network virtualization: a survey, some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 358–380, 2015.

[12] J. van de Belt, H. Ahmadi, and L. E. Doyle, "Defining and surveying wireless link virtualization and wireless network virtualization," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1603–1627, 2017.

[13] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 655–685, 2016.

[14] O. Soualah, N. Aitsaadi, and I. Fajjari, "A novel reactive survivable virtual network embedding scheme based on game theory," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 569–585, 2017.

[15] Z. Yan, J. Ge, Y. Wu, L. Li, and T. Li, "Automatic virtual network embedding: a deep reinforcement learning approach with graph convolutional networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1040–1057, 2020.

[16] H. Cao, L. Yang, and H. Zhu, "Novel node-ranking approach and multiple topology attributes-based embedding algorithm for single-domain virtual network embedding," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 108–120, 2018.

[17] N. Shahriar, R. Ahmed, S. R. Chowdhury et al., "Virtual network embedding with guaranteed connectivity under multiple substrate link failures," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1025–1043, 2020.

[18] Z. Zhao, X. Meng, Y. Su, and Z. Li, "Virtual network embedding based on node connectivity awareness and path integration evaluation," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 7, 2017.

[19] Y. Su, X. Meng, Z. Yu, and Q. Kang, "Cognitive virtual network topology reconfiguration method based on traffic prediction and link importance," *IEEE Access*, vol. 7, pp. 138915–138926, 2019.

[20] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding LC-VNE algorithms towards integrated node and link mapping," *IEEE/ACM Transactions on Networking*, vol. 24, no. 6, pp. 3648–3661, 2016.

[21] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *Journal of Lightwave Technology*, vol. 32, no. 3, pp. 450–460, 2014.

[22] H. Jiang, Y. Wang, L. Gong, and Z. Zhu, "Availability-aware survivable virtual network embedding in optical datacenter networks," *Journal of Optical Communications and Networking*, vol. 7, no. 12, 2015.

[23] M. Li, C. Hua, C. Chen, and X. Guan, "Application-driven virtual network embedding for industrial wireless sensor networks," in *IEEE International Conference on Communications (ICC)*, Paris, France, May 2017.

[24] D. Yun, J. Ok, B. Shin, S. Park, and Y. Yi, "Embedding of virtual network requests over static wireless multihop networks," *Computer Networks*, vol. 57, no. 5, pp. 1139–1152, 2013.

[25] R. Katona, V. Cionca, D. O'Shea, and D. Pesch, "Virtual network embedding for wireless sensor networks time-efficient QoS/QoI-aware approach," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 916–926, 2021.

[26] O. Kaiwartya, A. H. Abdullah, Y. Cao et al., "Virtualization in wireless sensor networks: fault tolerant embedding for internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 571–580, 2018.

[27] 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.

[28] W. Fan, P. Li, Z. Han et al., "Dynamic virtual network embedding of mobile cloud system based on global resources in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 8161–8174, 2021.

[29] C. K. Dehury and P. K. Sahoo, "DYVINE: fitness-based dynamic virtual network embedding in cloud computing," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1029–1045, 2019.

[30] H. Cao, S. Wu, G. S. Aujla, Q. Wang, L. Yang, and H. Zhu, "Dynamic embedding and quality of service-driven adjustment for cloud networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1406–1416, 2020.

[31] N. Pathak, S. Misra, A. Mukherjee, A. Roy, and A. Y. Zomaya, "UAV virtualization for enabling heterogeneous and persistent UAV-as-a-service," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6731–6738, 2020.

[32] 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 of Things Journal*, vol. 7, no. 7, pp. 5906–5921, 2020.

[33] M. Lu, Y. Gu, and D. Xie, "A dynamic and collaborative multi-layer virtual network embedding algorithm in SDN based on reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2305–2317, 2020.

[34] M. Li, C. Chen, C. Hua, and X. Guan, "Intelligent latency-aware virtual network embedding for industrial wireless networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7484–7496, 2019.

[35] G. Sun, H. Yu, V. Anand, and L. Li, "A cost efficient framework and algorithm for embedding dynamic virtual network requests," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1265–1277, 2013.

[36] Z. Zhang, S. Su, J. Zhang, K. Shuang, and P. Xu, "Energy aware virtual network embedding with dynamic demands: online and offline," *Computer Networks*, vol. 93, pp. 448–459, 2015.

[37] Z. Xu, W. Liang, and Q. Xia, "Efficient embedding of virtual networks to distributed clouds via exploring periodic resource demands," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 694–707, 2018.

[38] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 543–553, 2017.

[39] J. Pei, P. Hong, K. Xue, and D. Li, "Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2179–2192, 2019.

[40] Y. Liu, Y. Lu, X. Li, Z. Yao, and D. Zhao, "On dynamic service function chain reconfiguration in IoT networks," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10969–10984, 2020.

[41] K. Chen, N. Lv, S. Zhao, X. Wang, and J. 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.

[42] K. Chen, S. Zhao, and N. Lv, "Network monitoring information collection in the SDN-enabled airborne tactical network," *International Journal of Aerospace Engineering*, vol. 2018, Article ID 1940842, 20 pages, 2018.

[43] B.-N. Cheng, R. Charland, P. Christensen, L. Veytser, and J. Wheeler, "Evaluation of a multihop airborne IP backbone with heterogeneous radio technologies," *IEEE Transactions on Mobile Computing*, vol. 13, no. 2, pp. 299–310, 2014.

[44] R. Amin, D. Ripplinger, D. Mehta, and B.-N. Cheng, "Design considerations in applying disruption tolerant networking to tactical edge networks," *IEEE Communications Magazine*, vol. 53, no. 10, pp. 32–38, 2015.

[45] Z. Li and Y. Wu, "Smooth mobility and link reliability-based optimized link state routing scheme for MANETs," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1529–1532, 2017.

[46] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos, "Wireless sensor network virtualization: a survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 553–576, 2016.

[47] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: a survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[48] H. Cao, H. Hu, Z. Qu, and L. Yang, "Heuristic solutions of virtual network embedding: a survey," *China Communications*, vol. 15, no. 3, pp. 186–219, 2018.

[49] H. Cao, H. Zhu, and L. Yang, "Collaborative attributes and resources for single-stage virtual network mapping in network virtualization," *Journal of Communications and Networks*, vol. 22, no. 1, pp. 61–71, 2020.