ReViNE: Reinforcement Learning-Based Virtual Network Embedding in Satellite-Terrestrial Networks

Ilora Maity, *Member, IEEE*, Thang X. Vu, *Senior Member, IEEE*, and Symeon Chatzinotas, *Fellow, IEEE*

***Abstract*—This paper addresses the virtual network embedding (VNE) problem in integrated satellite-terrestrial networks (STNs). VNE consists of mapping virtual network functions (VNFs) in a service function chain (SFC) to physical nodes and mapping virtual links connecting the VNFs to physical links. Compared to terrestrial networks, VNE in STNs is challenging due to the movement of the non-geostationary orbit (NGSO) satellites and limited onboard processing resources. A static VNE strategy fails to efficiently address the diverse requirements of heterogeneous service requests in such a highly dynamic topology. In addition, existing solutions do not consider the capacity limitation and connectivity duration of inter-satellite links (ISLs) and ground-to-satellite links, which are essential parameters for deploying a VNE strategy in STNs. This work proposes a heuristic solution and reinforcement learning (RL)-based improved solution for the VNE scheme that can dynamically modify the existing VNF deployment strategy to maximize the average service acceptance rate and revenue. The RL agent selects a suitable VNE strategy for each service request considering the time-varying network topology and service's requirements. The proposed scheme increases the service acceptance ratio by 19.95% compared to the benchmark TS-MAPSCH.**

***Index Terms*—Software-Defined Networking (SDN), Virtual Network Embedding (VNE), Reinforcement Learning, Non-Terrestrial Networks, Satellite Constellation.**

I. Introduction

The next-generation satellite-terrestrial network (STN) aims to support services with diverse objectives [1]. However, traditional STN architectures are not sufficiently flexible to address these heterogeneous services' specific quality of service (QoS) demands. Virtual network embedding (VNE) is a recent paradigm that can add flexibility to STNs by defining multiple virtual networks over the same physical infrastructure. VNE involves mapping virtual network requests (VNRs) to the substrate network [2]. A VNR consists of one or more VNFs arranged in a serial processing order, referred to as a service function chain (SFC) [3]. VNE is a two-step process – (1) mapping each VNF to a physical node in the substrate network for processing and (2) defining a subgraph over the network topology, which refers to mapping the virtual links connecting each pair of VNFs to physical links [4]. VNE relies on network function virtualization (NFV) technology [5] that dissociates network functions from hardware infrastructure and deploys virtual network functions (VNFs) on network nodes for execution [3] [6]. Integrating software-defined networking (SDN) with STN also eases request-specific management by separating the control and data planes [7]. However, inappropriate deployment of VNFs may increase service completion duration and reduce the service acceptance ratio.

This work was supported in whole, or in part, by the Luxembourg National Research Fund, via project ASWELL (ref. C19/IS/13718904) and RUTINE (ref. C22/IS/17220888). For the purpose of open access, and in fulfilment of the obligations arising from the grant agreement, the author has applied a Creative Commons Attribution 4.0 International (CC BY 4.0) license to any Author Accepted Manuscript version arising from this submission.

A preliminary version of this work is published in IEEE WCNC 2022, Austin, TX, USA (April 10-13, 2022), DOI: 10.1109/WCNC51071.2022.9771560.

The authors are with the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg (Email: {ilora.maity, thang.vu, symeon.chatzinotas} @uni.lu).

A. Motivation

The topology of STN is highly dynamic due to the movement of non-geostationary orbit (NGSO) satellites. Accordingly, in STN, the link connectivity and quality change with time. This significantly impacts VNE, the primary mechanism for realizing SDN/NFV-based service provisioning in STN. A fixed VNE strategy is sufficient for terrestrial networks where the link connectivity is static over time. However, in STN, a fixed VNE strategy [5] [8] underperforms for several reasons. For example, the mapped physical link may become unavailable when a VNF needs to send the processed result to the next VNF in the SFC or the destination ground station. Moreover, the capacity of a mapped physical link may fluctuate due to time-varying topology. Therefore, for STN, dynamic VNE is necessary where an existing VNE strategy is modified, taking into account the network statistics and traffic demand. In this context, it is worth mentioning that several factors, such as link connectivity and capacity, and completion time requirements of heterogeneous VNRs, influence the selection of a dynamic VNE strategy. However, existing VNE solutions in STN do not consider these factors jointly [9] [1].

Figure 1 shows a motivational scenario with a substrate network consisting of three NGSO satellites (s_1, s_2, and s_3) and two ground stations (g_1 and g_2). The satellites have onboard computing resources [10]. At topology snapshot 1, a VNR with a single VNF is requested at g_1 whose destination is g_2. The routing path $g_1 \rightarrow s_1 \rightarrow s_2 \rightarrow g_2$ is selected for the VNR, and the VNF is mapped to s_2. Let us assume that the processing of the VNF is incomplete in the topology snapshot 2. From Figure 1, we see that in the topology snapshot 2, the link between s_2 and g_2 is unavailable. This may happen if s_2 moves out of the visibility range of g_2 or the link

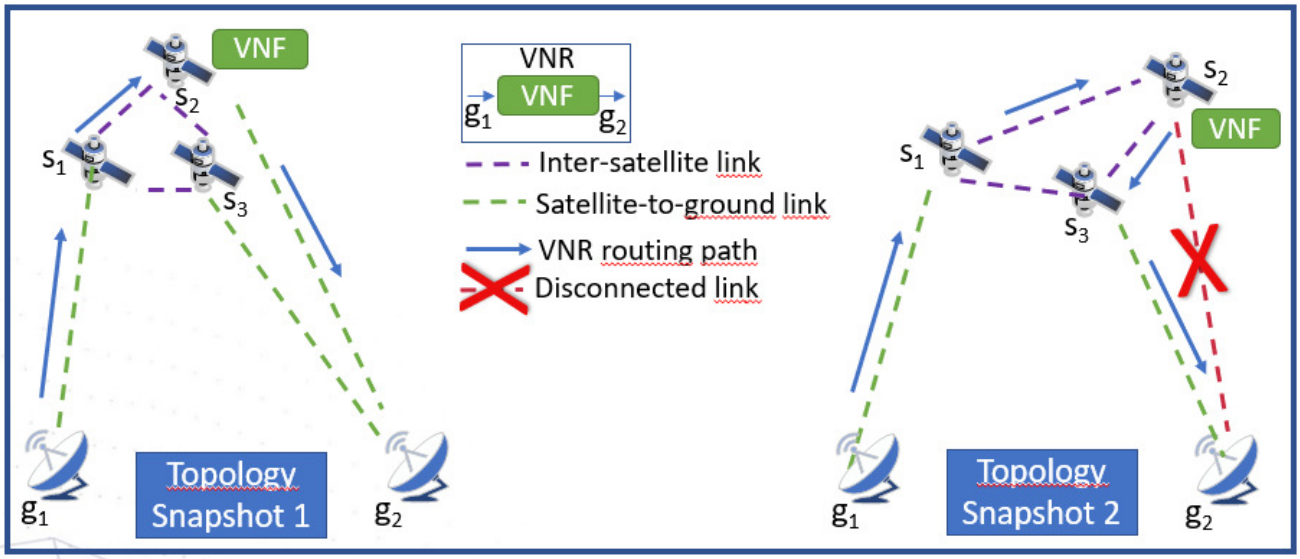

Fig. 1: Motivating Scenario

cannot accommodate the VNR, which requires a certain data rate. In this case, the routing path of the VNR is updated to $g_1 \rightarrow s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow g_2$. This routing path update incurs additional latency for the VNR. However, from Figure 1, we observe that the route $g_1 \rightarrow s_1 \rightarrow s_3 \rightarrow g_2$ is more stable than the route $g_1 \rightarrow s_1 \rightarrow s_2 \rightarrow g_2$. Therefore, mapping the VNF to s_3 and routing the VNR through the path $g_1 \rightarrow s_1 \rightarrow s_3 \rightarrow g_2$ is a feasible VNE strategy. This scenario intensifies in the presence the multiple heterogeneous VNRs with one or multiple VNFs. Motivated by this scenario, in this work, we consider link connectivity and stability as the metric for selecting the VNE strategy in STN.

The motivation scenario presented above is a simplistic example. However, in real networks, the scenario is complex due to the many satellites and several VNRs of different types. Traditional approaches for VNE rely on heuristic algorithms that often struggle to handle such networks' complex and dynamic nature. On the other hand, machine learning (ML) techniques, specifically reinforcement learning (RL), have demonstrated their capacity to adapt and optimize decision-making processes through environmental interactions [11] [12] [13] [14]. In the case of VNE, the training of RL algorithms can occur either online, through a real system, or offline, using a generative emulator representing the real environment. By leveraging RL algorithms, the VNE process can be enhanced by enabling intelligent, data-driven decision-making that adapts to the evolving network conditions and requirements. A single controller or a group of controllers can handle traffic engineering. These controllers serve as the operational entity where the RL-learned policies are applied during network operation. Using RL-learned policies at the controller level allows efficient and effective resource allocation, improved network performance, and enhanced scalability in STNs, ultimately contributing to the overall quality of service and user experience.

B. Contribution

In this work, we propose an RL-based dynamic VNE approach for STNs. The proposed solution, ReViNE, considers time-varying topology with NGSO satellites, processing resource limitation on satellite nodes, and capacity limitation of the substrate network as the parameters for VNE in STN. The proposed solution also facilitates VNF remapping to address the topology dynamics of STN. The primary contributions of this work are summarized as follows:

- We formulate the VNE problem based on the link and capacity availability of the substrate network and the processing time and data rate requirement of the VNRs.
- We design a linear relaxation-based greedy heuristic algorithm to decide the VNE strategy for new and existing VNRs.
- We design an RL-based VNE solution that adapts to the changing network conditions. The formulated reward function prioritizes the path with high revenue, low VNR deployment cost, high link stability, and high link capacity.
- Extensive simulation results show the effectiveness of ReViNE in terms of request acceptance ratio and service revenue.

C. Paper Organization

The remainder of the paper is organized as follows. Section II discusses the literature on VNE in terrestrial and non-terrestrial networks. Section III describes the system model in terms of the physical network model, virtual network model, and cost model. Section IV formulates the problem and presents the proposed heuristic solution and the RL framework. Section V evaluates the performance of the proposed scheme. Finally, Section VI concludes the paper.

II. Related Work

In this section, we discuss recent works on VNE in both terrestrial networks and non-terrestrial networks.

A. VNE in Terrestrial Networks

The allocation of resources to VNF and mapping VNFs to physical nodes are the main topics of existing works relating to VNE in terrestrial networks. A Deep Reinforcement Learning (DRL) model for the dynamic scaling of VNF resources was developed by Jalodia *et al.* [3]. Additionally, a Graph Neural Network (GNN) model was used in this study to quantify topological dependencies. Alhussein *et al.* [5] proposed a heuristic method for multicast routing and VNF mapping, which prioritizes service requests based on the request size and provisioning cost. In [15], resilient VNF placement was investigated to lessen the number of SFC requests impacted when a physical node fails. Liu *et al.* [8] designed heuristic algorithms for dynamic VNF placement and routing, considering resource and QoS constraints in terrestrial networks. The Dynamical VNF Placement and Routing Problem (DVPRP) and the Delay, packet Loss, and Jitter Aware Dynamical VNF Placement and Routing Problem (DLJA-DVPRP) are two optimization problems were developed therein. To meet the rule space's capacity, resource, and rate constraints, DVPRP seeks to minimize the total cost of links and nodes in the routing path of an SFC request and the cost of VNF deployment along the path. DLJA-DVPRP also considers additional restrictions such as end-to-end (E2E) delay, jitter, and packet loss.

B. VNE in Non-Terrestrial Networks

Several recent works on VNE have considered SDN/NFV-enabled non-terrestrial networks (NTNs). The authors of [16] put forth a framework with a topology awareness module to track changes to the network topology and link status in the Space-Ground Integrated Network (SGIN). The best routing strategy was then determined using a DRL model based on the most recent topology information. A reconfigurable VNF embedding strategy was presented by Wang *et al.* [1] to maximize the number of service requests that can be accommodated while minimizing computation and communication costs. Due to the higher resource availability in the ground network, it is preferred in this approach over the aerial network. However, this work does not consider the readjustment of the current VNF mapping and dynamic changes in network conditions. Moreover, if the node has sufficient resources, this approach assigns all VNFs for a service request to that node. In a real-world scenario, each node can process VNFs of a particular service type, such as [9]. The work of Li *et al.* [9] emphasizes the need for dynamic adjustment of VNF mapping and scheduling strategy to handle a range of Internet of Vehicle (IoV) service demands in a Space-Air-Ground Integrated Network (SAGIN). The authors addressed the E2E delay requirement of recently received service requests by proposing a Tabu search-based VNF remapping and rescheduling (TS-MAPSCH) algorithm that modifies the current VNF mapping and scheduling strategy. This approach does not explicitly consider the effects of changes in link properties for a dynamic network topology like satellite networks. This method also does not contemplate the rate constraints of physical links. Yuan *et al.* [17] proposed a greedy algorithm for joint VNF placement and routing in STNs aiming to minimize the average E2E service latency. This approach approximates the dynamics of STN topology with a time-evolving graph (TEG). However, this work does not consider VNF deployment costs, including the VNF remapping overhead. Yang *et al.* [18] also exploit TEG for joint VNF placement and flow routing in space information networks. This work considers a full cooperation VNF deployment scheme where the same VNF can be deployed at multiple nodes accessible by each VNR. Therefore, this work aims to maximize the network flow by balancing the coordination overhead from deploying the same VNF at multiple nodes. Yue *et al.* [19] investigate VNF placement problem in 6G NTN. The authors formulated the VNF placement problem as an optimal matching problem and proposed a Linear Programming (LP)-based approach to minimize the distance between the weighted graphs representing the SFC requests and the physical network. Subsequently, this work designs a Hungarian-based algorithm for SFC mapping, considering the delay and resource requirements of each SFC.

C. Learning-Based VNE

Some current works use AI for resource management based on VNF mapping. These studies take into account terrestrial networks, though. Deep learning is used, for instance, in [20] to map VNFs and allocate processing resources to the VNFs. The time-varying nature of the traffic load is taken into account in this work. Another study [21] uses DRL actor-critic architecture to allocate processing and storage resources to VNFs for 5G terrestrial networks. The authors first develop a Markov decision process (MDP) for dynamic resource allocation. The deployed VNFs are then online resource-allocated by a DRL-based algorithm following their dynamic resource needs. Also, DRL is used in related work [22] to distribute data center (DC) resources among VNFs.

The analysis of the existing literature reveals that, except for the strategy suggested in [9], most works concentrate on static VNE. Our work considers dynamic VNF mapping, where the current strategy changes depending on the situation, similar to [9]. Unlike the existing literature, our work considers additional restrictions on the strength and stability of physical links, such as the duration of link availability and capacity. We also consider an NGSO constellation as the substrate network, where the availability of multiple E2E routes adds complexity to the VNE process, which is relevant in the real-world scenario. Moreover, incorporating an RL-based solution adds a layer of adaptability to the VNE scheme. The RL agent dynamically selects a suitable VNE strategy for each VNR, considering (1) VNR-specific parameters such as source and destination nodes, revenue, latency requirements, and deployment cost and (2) network-specific parameters such as available E2E paths, and stability and capacity of the links on each path.

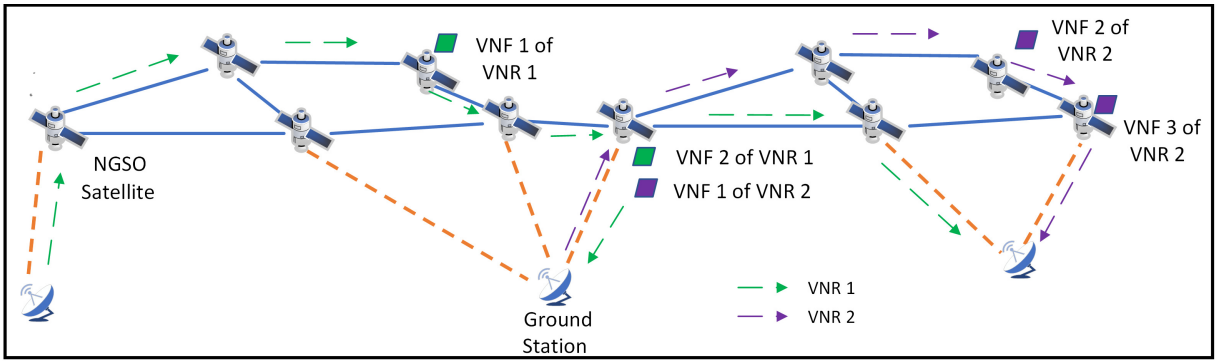

Fig. 2: Network Architecture

III. System Model

A. Physical Network Model

As shown in Figure 2, we consider a software-defined STN consisting of a set of nodes $\mathcal{N}$ in the data plane, which includes a set of low earth orbit (LEO) satellite nodes $\mathcal{N}_s$ and a set of ground nodes $\mathcal{N}_g$. The ground and satellite nodes vary in terms of processing capacity [23] [24]. Let p_i denote the processing capacity of n_i in terms of processing units. Let $\mathcal{E}$ denote the set of links and $e_{ij} \in \mathcal{E}$ represent the link connecting $n_i \in \mathcal{N}$ and $n_j \in \mathcal{N}$. The links connected to satellite nodes are time-varying regarding availability, link capacity, and propagation latency. The time-varying network topology is described via multiple time slots where the network topology is static in each time slot [25]. Let the duration of each time slot be σ time units and the start time of the t^{th} time slot be t^{start}. Let $b_{ij}[t]$ and $\delta_{ij}[t]$ denote the available capacity in Mbps and propagation latency of e_{ij}, respectively.

***Definition* 1** (Link Connectivity)**.** *The link connectivity is indicated as:*

$$a_{ij}[t] = \begin{cases} 1 & \textit{only if the link } e_{ij} \textit{ is available at time slot } t, \\ 0 & \textit{otherwise}. \end{cases} \tag{1}$$

Factors such as satellite visibility, minimum elevation angle, and antenna re-pointing periods determine link connectivity.

***Definition* 2** (Link Stability)**.** *The link stability is expressed as:*

$$s_{ij}[t] = \begin{cases} m & \textit{if } a_{ij}[t] = 1 \textit{ and } e_{ij} \textit{ connected for } m \textit{ slots}, \\ & \textit{starting from the current time slot } t, \\ 0 & \textit{otherwise}. \end{cases} \tag{2}$$

Table I depicts the list of symbols used in this paper.

B. Virtual Network Model

Let $\mathcal{F}[t]$ denote the set of VNRs that are new or currently in process at time slot t. Each VNR consists of an ordered set of VNFs installed and executed on network nodes. User Equipment (UE) associated with ground nodes request these VNRs. The request is unidirectional and can be either point-to-point or point-to-multipoint.

A VNR $f_k \in \mathcal{F}[t]$ is represented by a tuple $< n_s^k, \mathcal{N}_d^k, V^k, \mathcal{E}^k, t_{arr}^k, r^k, D^k >$, where $n_s^k \in \mathcal{N}$ is the source node, $\mathcal{N}_d^k \subset \mathcal{N}$ is the set of destination nodes, V^k is an ordered set of VNFs, $\mathcal{E}^k$ is the set of virtual links connecting the VNFs, t_{arr}^k is the arrival time of the request, r^k is the normalized revenue for servicing the request, and D^k is the maximum allowable delay.

We consider that a VNF $v_a \in V^k$ requires β_a processing units. A virtual link $e_{gh} \in \mathcal{E}^k$ has rate unit requirement denoted by b_{gh}. Each virtual link is mapped to one or multiple physical links. The following binary variable indicates whether a virtual link is mapped to a physical link at time slot t:

$$y_{gh}^{ij}[t] = \begin{cases} 1 & \text{if } e_{gh} \in \mathcal{E}^k \text{ is mapped to } e_{ij} \in \mathcal{E} \text{ at } t, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The mapping of a VNF v_a to a node n_i at time slot t is denoted by the following binary variable:

$$x_i^a[t] = \begin{cases} 1 & \text{if } v_a \text{ is mapped to } n_i \text{ at } t, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The processing duration of f_k is estimated as:

$$d_k = \sum_{v_a \in V^k} \sum_{n_i \in \mathcal{N}} x_i^a[t] \delta_i^a + \sum_{e_{gh} \in \mathcal{E}^k} \sum_{e_{ij} \in \mathcal{E}} y_{gh}^{ij}[t] \delta_{ij}[t], \tag{5}$$

where δ_i^a is the execution time for v_a at n_i. Therefore, the number of time slots required to process f_k is $t_k^{req} = \lceil \frac{d_k}{\sigma} \rceil$.

C. Cost Model

The VNR deployment cost consists of the cost of embedding new VNRs, which may involve the migration of VNF mapping for incomplete requests arrived at earlier time slots [26]. This cost depends on the number of VNFs mapped or remapped for a given service. For an existing VNR f_k where $t^{k,arr} < t^{start}$, the VNF deployment cost includes the cost of remapping already mapped VNFs and is given by:

$$C_k^d[t] = \frac{\sum_{v_a \in V^k} \sum_{n_i \in \mathcal{N}} x_i^a[t] \left(1 - x_i^a[t-1]\right) \left(c_r^i + c_m^i\right)}{\sum_{n_i \in \mathcal{N}} \left(c_r^i + c_m^i\right)}, \tag{6}$$

where c_m^i and c_r^i are the costs in time units for instantiating and remapping a VNF to n_i, respectively. For a new VNR arriving at the current slot, the VNF deployment cost is given by:

$$C_k^d[t] = \frac{\sum_{v_a \in V^k} \sum_{n_i \in \mathcal{N}} x_i^a[t] c_m^i}{\sum_{n_i \in \mathcal{N}} \left(c_r^i + c_m^i\right)}. \tag{7}$$

TABLE I: Table of Symbols

Symbol	***Definition***	***Measurement Unit (if applicable)***
$\mathcal{N}$	Set of physical nodes	
$\mathcal{N}_s$	Set of LEO satellites	
$\mathcal{N}_g$	Set of ground nodes	
p_i	Processing capacity of $n_i \in \mathcal{N}$	T flops
$\mathcal{E}$	Set of links	
σ	Duration of a time slot	seconds
t^{start}	Start time of slot t	
$b_{ij}[t]$	Available capacity of $e_{ij} \in \mathcal{E}$	Mbps
$\delta_{ij}[t]$	Propagation latency of $e_{ij} \in \mathcal{E}$	milliseconds
$\mathcal{F}[t]$	Set of VNRs	
n_s^k	Source ground node of $f_k \in \mathcal{F}[t]$	
$\mathcal{N}_d^k$	Set of destination ground nodes of $f_k \in \mathcal{F}[t]$	
V^k	Ordered set of VNFs for $f_k \in \mathcal{F}[t]$	
$\mathcal{E}^k$	Set of virtual links for $f_k \in \mathcal{F}[t]$	
t_{arr}^k	Arrival time of $f_k \in \mathcal{F}[t]$	
r^k	Normalized service revenue of $f_k \in \mathcal{F}[t]$	
D^k	Maximum allowable delay of $f_k \in \mathcal{F}[t]$	milliseconds
β_a	Processing units required by a VNF $v_a \in V^k$	
b_{gh}	Data rate required by a virtual link $e_{gh} \in \mathcal{E}^k$	Mbps
δ_i^a	Execution time of $v_a \in V^k$ at $n_i \in \mathcal{N}_s$	milliseconds
c_m^i	Cost of instantiating a VNF at $n_i \in \mathcal{N}_s$	milliseconds
c_r^i	Cost of remapping a VNF to $n_i \in \mathcal{N}_s$	milliseconds

***Definition* 3** (Service Revenue)**.** *The revenue for serving an SFC request f_k is given by:*

$$I_k = r^k \frac{D^k - d_k}{D^k}. \tag{8}$$

A VNR is only accepted if the estimated completion duration, as mentioned in Equation (5), is less than the respective maximum allowable delay. Therefore, we use a binary variable A to denote the acceptance of a VNR. Mathematically,

$$A_k[t] = \begin{cases} 1 & \text{if } d_k \leq D^k, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

***Definition* 4** (Request Acceptance Ratio)**.** *The request acceptance ratio at time slot t is given by:*

$$AR[t] = \frac{\sum\limits_{f_k \in \mathcal{F}[t]} A_k[t]}{|\mathcal{F}[t]|}. \tag{10}$$

IV. Dynamic VNE for STN

We consider that at any time slot, VNRs arrive dynamically. Therefore, some of these requests may continue over the next time slot. This work aims to construct a dynamic VNF mapping strategy to minimize total cost and maximize revenue considering the effects of changes in the network topology and available resources.

A. Problem Formulation

The work aims to maximize the revenue and minimize the VNF deployment and migration cost at a time slot t. The decision variables are $x[t]$ and $y[t]$, which denote the mappings of VNFs and virtual links, respectively. Mathematically,

$$\textbf{P0:} \underset{x[t],y[t]}{\text{Maximize}} \sum_{f_k \in \mathcal{F}[t]} I_k - C_k^d[t] \tag{11}$$

subject to

$$\sum_{n_i \in \mathcal{N}} x_i^a[t] = 1, \forall v_a \in V_k, \forall f_k \in \mathcal{F}[t], \tag{12}$$

$$\sum_{f_k \in \mathcal{F}[t]} \sum_{v_a \in V_k} x_i^a[t]\beta_a \leq p_i, \forall n_i \in \mathcal{N}, \tag{13}$$

$$y_{gh}^{ij}[t] \leq a_{ij}[t], \forall e_{gh} \in E^k, \forall f_k \in \mathcal{F}[t], \forall e_{ij} \in \mathcal{E}, \tag{14}$$

$$\sum_{f_k \in \mathcal{F}[t]} \sum_{e_{gh} \in \mathcal{E}} y_{gh}^{ij}[t] b_{gh} \leq b_{ij}[t], \forall e_{ij} \in \mathcal{E}, \tag{15}$$

$$\sum_{n_i \in \mathcal{N}} y_{gh}^{ij}[t] - \sum_{n_i \in \mathcal{N}} y_{gh}^{ji}[t] = x_j^h[t] - x_j^g[t],$$
$$\forall n_j \in \mathcal{N}, \forall e_{gh} \in \mathcal{E}^k, \forall f_k \in \mathcal{F}[t], \tag{16}$$

$$x_i^a[t] \in \{0,1\}, \forall n_i \in \mathcal{N}, \forall v_a \in V_k, \forall f_k \in \mathcal{F}[t], \tag{17}$$

$$y_{gh}^{ij}[t] \in \{0,1\}, \forall e_{ij} \in \mathcal{E}, \forall e_{gh} \in \mathcal{E}^k, \forall f_k \in \mathcal{F}[t], \tag{18}$$

Equation (12) states that a VNF can be mapped to exactly one node. Equation (13) conveys that a satellite should have sufficient processing capacities to handle the mapped VNFs. The constraint mentioned in Equation (14) states that a VNR is routed through a link only if it is available. Equation (15) expresses the link capacity constraint. Equation (16) states the flow conservation constraint required to establish a VNR routing path.

The optimization problem formulated in Equation (11) is an Integer Linear Programming (ILP) problem with binary decision variables $x[t]$ and $y[t]$, which is computationally

expensive for large-scale networks with high number of NGSO satellites and equivalently high number of VNRs. After relaxing the binary variables as continuous variables, problem P0 can be re-written as follows:

$$\textbf{P1: Maximize}_{x[t],y[t]} \sum_{f_k \in \mathcal{F}[t]} I_k - C_k^d[t] \quad (19)$$

subject to

$(12-16)$,

$$0 \leq x_i^a[t] \leq 1, \forall n_i \in \mathcal{N}, \forall v_a \in V_k, \forall f_k \in \mathcal{F}[t], \quad (20)$$

$$0 \leq y_{gh}^{ij}[t] \leq 1, \forall e_{ij} \in \mathcal{E}, \forall e_{gh} \in \mathcal{E}^k, \forall f_k \in \mathcal{F}[t], \quad (21)$$

Algorithm 1 Dynamic VNE (D-ViNE) Algorithm

INPUT: P1
OUTPUT: $\{x[t], y[t]\}$
PROCEDURE:

Solve the relaxed optimization problem P1 in Equation (19) to obtain solution $\{x1[t], y1[t]\}$
for each $x1_i^a[t] \in x1[t]$ **do**
  if $x1_i^a[t] \in \mathbb{Z}$ **then** ▷ Integer solution
    $x_i^a[t] \leftarrow x1_i^a[t]$
  else
    $\Lambda^n \leftarrow \{\phi\}$
    for all $n_i \in \mathcal{N}$ **do**
      $n_i \leftarrow \underset{n_i \in \mathcal{N} \setminus \Lambda^n}{\text{argmax}}\ x1_i^a[t]$, $\Lambda^n \leftarrow \Lambda^n \cup \{n_i\}$
      if constraint (13) is not violated **then**
        Set $x_i^a[t] \leftarrow 1$
        break
      end if
    end for
  end if
end for
for each $y1_{gh}^{ij}[t] \in y1[t]$ **do**
  if $y1_{gh}^{ij}[t] \in \mathbb{Z}$ **then** ▷ Integer solution
    $y_{gh}^{ij}[t] \leftarrow y1_{gh}^{ij}[t]$
  else
    $\Lambda^e \leftarrow \{\phi\}$
    for all $e_{ij} \in E$ **do**
      $e_{ij} \leftarrow \underset{e_{ij} \in E \setminus \Lambda^e}{\text{argmax}}\ y1_{gh}^{ij}[t]$, $\Lambda^e \leftarrow \Lambda^e \cup \{e_{ij}\}$
      if constraints (14-16) are not violated **then**
        Set $y_{gh}^{ij}[t] \leftarrow 1$
      end if
    end for
  end if
end for
return $\{x[t], y[t]\}$

B. Heuristic Solution

Based on the solution of P1, we propose a heuristic algorithm to generate a feasible solution. Algorithm 1 shows the Dynamic VNE (D-ViNE).

D-ViNE algorithm considers the relaxed optimization problem P1 as an input and generates the feasible solution represented by the set of decision variables $x[t], y[t]$. Line 1 computes the solution of P1 denoted by the set of decision variables $x1[t], y1[t]$. We use CVX solver [27] to solve P1. D-ViNE algorithm performs VNF mapping and virtual link mapping sequentially. For VNF mapping, if a decision variable $x1_i^a[t] \in x1[t]$ is an integer, D-ViNE algorithm directly maps $x1_i^a[t]$ to $x_i^a[t] \in x[t]$. Otherwise, we select an unvisited node $n_i \in N$ for which the value of $x1_i^a[t]$ is the highest and check whether mapping the VNF v_a to n_i satisfies the constraint mentioned in Equation (13). If the constraints are satisfied, we set $x_i^a[t] \leftarrow 1$. This process continues until a node where the respective VNF can be mapped is found. The set Λ^n contains the set of visited nodes. Similarly, for virtual link mapping, if a decision variable $y1_{gh}^{ij}[t] \in y1[t]$ is an integer, D-ViNE algorithm maps $y1_{gh}^{ij}[t]$ to $y_{gh}^{ij}[t] \in y[t]$. Otherwise, we select an unvisited edge $e_{ij} \in E$ for which the value of $y1_{gh}^{ij}[t]$ is the maximum and check whether mapping virtual link $e_{gh} \in E^k$ to physical link e_{ij} satisfies the constrains mentioned in Equations (14-16). If the constraints are satisfied, we set $y_{gh}^{ij}[t] \leftarrow 1$. The set Λ^e contains the visited physical links.

The computational complexity of Algorithm 1 depends on the time required for mapping the nodes and links. For node mapping, Steps 2-15 takes $O(|\mathcal{N}|^2 \sum_{f_k \in \mathcal{F}} |\mathcal{V}_k|)$ time. For link mapping, Steps 16-28 takes $O(|\mathcal{E}|^2 \sum_{f_k \in \mathcal{F}} |\mathcal{E}_k|)$. Therefore, the computational complexity of Algorithm 1 is $O(|\mathcal{N}|^2 \sum_{f_k \in \mathcal{F}} |\mathcal{V}_k| + |\mathcal{E}|^2 \sum_{f_k \in \mathcal{F}} |\mathcal{E}_k|)$ which can be approximated as $O(|E|^2)$ for large-scale networks.

C. ReViNE: RL-Based VNE

We propose an RL-based algorithm that offers an adaptive and optimal solution compared to the heuristic solution. We define the deep Q network (DQN) framework where an RL agent aims to maximize the revenue and minimize the cost at t for each unfinished VNR f_k that arrived in the current or earlier time slot. DQN is a promising algorithm for high-volume data [28], which is evident in the network scenario considered in our work, where the number of nodes (LEO satellites and ground) is large.

1) DQN Framework: The RL model is represented as a Markov decision process (MDP) defined by the tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the set of transition probabilities, and $\mathcal{R}$ is the reward function. For the dynamic VNE scheme, the MDP parameters are defined as follows:

a) State Space: The state space (S) at time τ refers to the maximum available processing units in each path between the source and destination nodes. Therefore, we have:

$$\mathcal{S} = \{p^{i,\tau}\}, \forall path_i \in Paths, \quad (22)$$

where $p^{i,\tau}$ denotes the maximum available processing units of the i^{th} path, and $Paths$ denote the set of all paths between a source-destination pair at time τ.

b) Action Space: The action space at time τ refers to selecting the path for mapping the VNRs. VNFs are mapped according to their order in the VNR to satellites on the selected path with sufficient processing units available, starting with the

satellite having maximum residual processing units. Therefore, the action space $\mathcal{A}$ at time τ is the set of common paths between the source node n_k^s and all of the destination nodes in $\mathcal{N}_k^d$. The link mapping y is computed based on mapped physical nodes. For example, if $x_i^a[t] = 1$ and $x_j^{a+1}[t] = 1$, $y_{gh}^{ij}[t] = 1$ where e_{gh} is the virtual link between $v_{a,}$ and v_{a+1}.

c) Reward Function: The reward function $r_\tau \in \mathcal{R}$ is its progress towards the goal of the optimization problem P1 which is heuristically inspired. For the composition of the reward function, we consider the following parameters for the formulated problem P1:

- Service Revenue: As mentioned in P1, the objective is to maximize the service revenue. Therefore, we select the service revenue as a component of the reward function. The reward increases as the service revenue increases, which signifies processing all VNFs fast and within the maximum allowable delay of the VNR.
- VNR Deployment Cost: P1 aims to minimize the VNR deployment cost. Therefore, we select the VNR deployment cost as a component of the reward function. The VNR deployment cost increases when VNFs are remapped to meet the service demand. However, the reward function ensures that the number of remapping is limited as the reward decreases with the VNR deployment cost.
- Link Stability: As mentioned in Constraint (14) of P1, the mapped physical link should be available during the lifetime of the VNRs. Therefore, we select link stability as a parameter of the reward function. It assists the agent in preferring more stable physical links for mapping the virtual links. The minimum link stability on the E2E path selected to route a VNR f_k is given by:
$$s_k^{min} = \min_{e_{gh} \in \mathcal{E}^k, y_{gh}^{ij}[t]=1, e_{ij} \in \mathcal{E}} (s_{ij}[t]) \tag{23}$$
- Link Capacity: As mentioned in Constraint (15) of P1, a physical link should have sufficient capacity to accommodate all the virtual links mapped to it. Therefore, we select the surplus link capacity available on the links in the E2E path as a parameter of the reward function. Intuitively, the reward increases with the residual link capacity.

Mathematically,

$$r_\tau = \frac{1}{|\mathcal{F}^{'}[t]|} \sum_{f_k \in \mathcal{F}^{'}[t]} (\alpha_1 I_k - \alpha_2 C_k^d[t] + \alpha_3 \frac{s_k^{min} - t_k^{req}}{s_k^{min}} + \alpha_4 \sum_{e_{gh} \in E^k} \sum_{e_{ij} \in E} y_{gh}^{ij}[t] \frac{b_{ij}[t] - b_{gh}}{b_{ij}[t]}), \tag{24}$$

where $\mathcal{F}^{'}[t] \subset \mathcal{F}[t]$ is the set of VNRs between the pair of source and destination nodes, $\alpha_i, i \in \{1, 2, 3, 4\}$ are the weight coefficients, and t_k^{req} is the slots required to process f_k.

In DQN, the Q-values or the probability of taking actions are estimated to optimize the reward [29]. We use DQN to learn a parametrized value function $Q(s, a; \theta)$ that approximates the optimal Q-values. The value function $Q(s_\tau, a; \theta_u)$ is obtained based on the one-step look ahead $r_\tau + \gamma \max_a Q(s_{\tau+1}, a; \theta_u)$, where γ is the discount factor. Hence, the parameter θ_u determines $Q(s_\tau, a; \theta_u)$, which in turn influences the selection of a good action. The loss function is given by:

$$\mathcal{L}(\theta_u) = \begin{cases} (r_\tau + \gamma \max_a Q(s_{\tau+1}, a; \theta_u) \\ -Q(s_\tau, a; \theta_u))^2 & \text{for non-terminal state,} \\ r_\tau & \text{otherwise,} \end{cases} \tag{25}$$

DQN aims to determine the optimal θ_u^* that minimizes $\mathcal{L}(\theta_u)$.

Algorithm 2 Dynamic VNE Algorithm

INPUTS: γ, ϵ, M_e, T, $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$
OUTPUT: Optimal policy, $x[t]$, $y[t]$
PROCEDURE:
1: Initialize replay buffer B
2: Initialize Q-network with parameters $\theta_u \leftarrow \theta$
3: Initialize target network with parameter θ_u
4: **for** $episode = 1$ to M_e **do**
5: Reset the environment and initialize state s_τ
6: **for** $\tau = 1$ to T **do**
7: With probability ϵ select a random action a_τ
8: Otherwise, select action $a_\tau = \arg\max Q(s, a)$
9: Execute action a_τ in the environment and receive reward r_τ and new state $s_{\tau+1}$
10: Store transition $(s_\tau, a_\tau, r_\tau, s_{\tau+1})$ in replay buffer B
11: Set $s_\tau \leftarrow s_{\tau+1}$
12: Sample a random mini-batch of transitions $(s_\tau, a_\tau, r_\tau, s_{\tau+1})$ from B
13: Compute loss function using (25)
14: Perform gradient descent for the Q-network and update the parameters θ_u
15: **end for**
16: Reduce ϵ according to the decay schedule
17: **end for**
18: Update node mapping $x[t]$ and link mapping $y[t]$ based on the optimal policy

Algorithm 2 presents the RL-based dynamic VNE procedure. Lines 1-3 initialize the replay buffer B and parameters for the Q-network and target network, respectively. The training is performed for M_e episodes, each with T steps. At the beginning of each episode, the environment is reset in Line 5. In each step, an action is selected and executed, and the transition is stored in B, the loss is computed using (25), and parameters of the Q-network are updated using gradient descent (Lines 6-15). The output of the learned Q-network is the optimal policy referring to optimal actions, which define the routing path of the VNRs in terms of node mapping $x[t]$ and link mapping $y[t]$. In this work, we do not consider VNF scheduling, i.e., determining the execution time of the VNFs mapped to the same node due to limited scope. However, the proposed VNE solution can be integrated with our work [30] on VNF scheduling for non-terrestrial networks.

The computational complexity of Algorithm 2 depends on the number of iterations, hidden layers, and neurons in each layer. The number of iterations depends on the number of episodes and the number of steps in each episode. Therefore, the total number of iterations is $M_e T$. Steps 7-14 take $O(H\eta^2)$ time, where H is the number of hidden layers and η is

the number of neurons in each layer [31]. Therefore, the computational complexity of Algorithm 2 representing a single DQN is $O(M_e T H \eta^2)$. Moreover, there are $|\mathcal{N}|(|\mathcal{N}|-1)$ source-destination pair which implies $|\mathcal{N}|(|\mathcal{N}|-1)$ DQNs. However, in reality, VNRs do not exist between all source-destination pairs at a time slot, which reduces the number of required DQNs. The computational complexity of each DQN can be further reduced by careful selection of the parameters M_e, T, H, and η^2.

TABLE II: Simulation Parameters

Parameter	Value
Satellite constellation	Iridium NEXT [32]
Inclination	86.4° [32]
Number of orbital planes	6 [32]
Constellation altitude	780 km [32]
Number of time slots	1441
Duration of a time slot (σ)	60 seconds
Number of LEO satellites ($\lvert\mathcal{N}_s\rvert$)	75 [32]
Number of ground nodes ($\lvert\mathcal{N}_g\rvert$)	70
VNR arrival rate	0.05-0.2 VNRs/s [9]
Processing capacity of a satellite node (p_i)	$20-40$ T flops [33]
Capacity of ISLs	$100-150$ Mbps [34]
Capacity of ground-to-satellite links	$150-200$ Mbps [34]
Maximum allowable delay of a VNR (D^k)	$100-125$ ms [9]
Number of VNFs in a VNR ($\lvert V_k\rvert$)	$3-5$ [9]
Revenue of a VNR	$50-100$ [9]
VNF execution time at a satellite node (δ_i^a)	$5-10$ ms [9]
Processing units required by VNF (β_a)	$10-20$ [9]
Rate required by virtual link (b_{gh})	$10-30$ Mbps [35]
VNF remapping cost (c_r^i)	4 ms [9]
VNF instantiation cost (c_m^i)	3 ms [9]
Weight coefficient (α)	0.25

V. Performance Evaluation

A. Simulation Settings

For the performance evaluation of the proposed scheme, we consider the Iridium NEXT constellation, which has 75 LEO satellites distributed in 6 orbital planes at 780 km altitude [32]. The constellation is simulated using MATLAB Satellite Communication Toolbox [36] and the two-line element (TLE) data of Iridium NEXT constellation for the date 21st October 2022 [32]. The total simulation duration of one day is divided into 1441 slots, each having a duration of 1 minute. Additionally, we consider 70 ground nodes placed optimally based on the work by Guo *et al.* [37]. VNRs are randomly generated between pair of nodes to ensure diversity. Table II states the simulation parameters. We assume that the network resources are sufficient to support all the VNRs, i.e., there exists a feasible solution to problem P0.

TABLE III: DQN Parameters

Parameter	Value
Discount factor (γ)	0.99
Epsilon (ϵ)	0.9
Epsilon decay	0.01
Maximum number of episodes (M_e)	500
Maximum number of steps per episode (T)	50

B. Simulation of the RL Model

The state and action spaces are defined based on the routes between the source and destination nodes for simulating the RL model. The learning algorithm is executed for a maximum of 500 episodes, each containing 50 learning steps. Each episode starts with an initial state which signifies the processing resource availability on the routes. Accordingly, the agent takes action by selecting a route, observes the resulting states and rewards, and learns from these experiences to update its policy. The DQN algorithm can handle a large state space due to multiple paths between source and destination nodes. For the design of the DQN, the input state is followed by two fully connected layers, each consisting of 64 units, consisting of the ReLU activation function. Table III specifies the parameters considered for the DQN model.

C. Benchmark Schemes

For performance evaluation, we consider two benchmark schemes – fixed VNE and TS-MAPSCH [9]. The fixed VNE scheme greedily maps VNFs to physical nodes based on the available processing units and does not facilitate VNF remapping. On the other hand, TS-MAPSCH examines the feasibility of all neighboring solutions for VNF mapping and selects the best one. TS-MAPSCH also remaps the VNFs as needed. Additionally, we compare the performance of the proposed DQN-based ReViNE solution with Policy Gradient (PG) that uses the same RL model formulated in IV-C1. We select PG as the benchmark because it is a model-free RL algorithm similar to DQN that can work well in discrete action space.

D. Performance Metrics

We consider the following metrics to evaluate the performance of ReViNE:

- **Request Acceptance Ratio**: This metric, as defined in Definition 4, quantifies the number of mapped VNRs estimated to complete within the maximum allowable delay based on a selected VNE strategy.
- **Average Service Revenue**: This metric is defined in Definition 3, and it measures the average profit for serving the requests based on the respective revenue of the requests.
- **VNF Deployment Cost**: As stated in Equations (6) and (7), we need this metric to evaluate the combined cost of VNF mapping and remapping to show the efficacy of the proposed dynamic VNE strategy.

- **Rate of VNF Remapping**: This metric quantifies the number of remapped VNFs to analyze the stability of the initial VNE strategy.

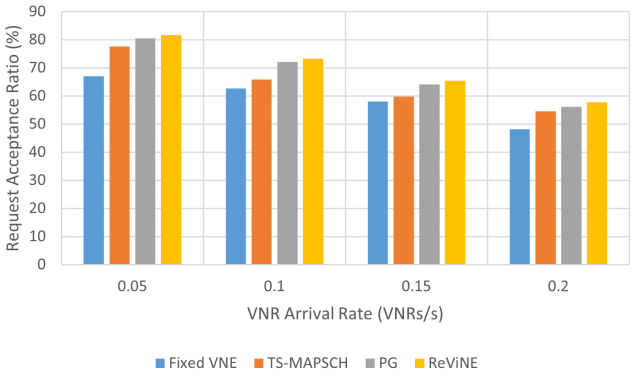

Fig. 3: Request Acceptance Ratio

E. Result and Discussion

1) Request Acceptance Ratio: Figure 3 shows the request acceptance ratio for different traffic volumes. From the simulation results, we observe that ReViNE accepts more requests compared to the benchmarks. In particular, for the VNR arrival rate of 0.2 VNRs/s, ReViNE accepts 6.09% and 19.95% more VNRs than the fixed VNE and TS-MAPSCH strategies. According to the reward function defined in (24), ReViNE maps each VNR to a stable routing path. Therefore, the VNRs are less affected by the topology changes and complete processing of the VNFs within the respective delay bound. Also, remapping unfinished requests increases the acceptance ratio. The request acceptance ratio for the fixed VNE strategy is low as it does not support VNF remapping, and some of the mapped VNRs underperform as the traffic volume increases. Also, we observe that all the schemes exhibit good performance for low traffic volume. This is because the available computation and link capacity resources are adequate for the services arriving at a low rate.

2) Average Service Revenue: As shown in Figure 4, the average service revenue for ReViNE is 4.17% and 34.93% more than fixed VNE and TS-MAPSCH for an arrival rate of 0.05 VNRs/s. This is because the RL agent maximizes the revenue of each VNR by mapping it to a path with the minimum E2E delay as stated in the reward function formulated in (24). We also observe that the performance gap between ReViNE and the benchmarks increases with the traffic volume. This is because the number of requests violating the latency demand rises with the traffic volume. In this case, ReViNE rearranges the mapping strategy more effectively than the benchmarks to guarantee that more VNRs complete processing within the respective delay bound. To be precise, ReViNE selects a stable path with low E2E delay for remapping existing VNRs. In comparison, the benchmark TS-MAPSCH ignores the link characteristics and facilitates VNR remapping only considering the slackness value, which is the difference between the respective delay demand and the actual E2E delay.

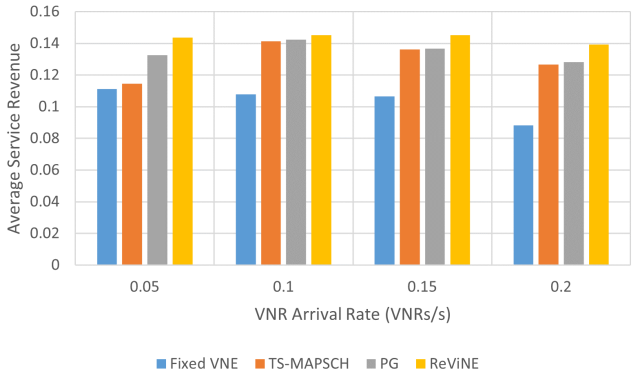

Fig. 4: Average Service Revenue

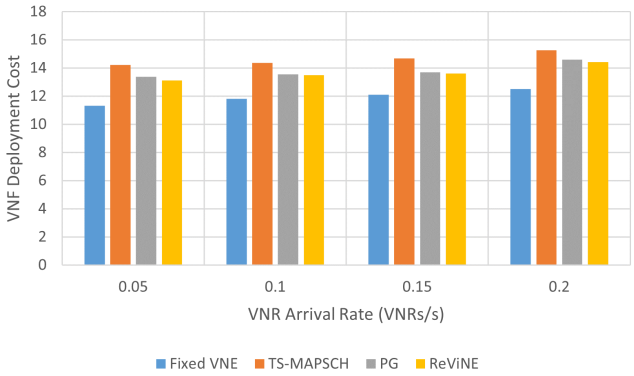

Fig. 5: VNF Deployment Cost

3) VNF Deployment Cost: We analyze the performance of the proposed scheme in terms of the deployment cost for mapping and remapping VNRs. Excessive remapping increases deployment cost, which refers to an inefficient VNE strategy for resource-constrained STNs and reduces the service provider's profit. Figure 5 shows that the deployment cost for the fixed VNE strategy is the minimum as it involves only the cost of mapping the VNFs for the first time without any provision for remapping. In addition, we observe that the VNF deployment cost of TS-MAPSCH is 5.74% more than ReViNE. This is because TS-MAPSCH redeploys the VNFs based on profit without considering the link status. On the other hand, ReViNE performs minimal VNF redeployment by mapping the VNFs considering link stability and capacity. Therefore, the VNF deployment cost for ReViNE is less than that of TS-MAPSCH.

4) Rate of VNF Remapping: Figure 6 shows that the benchmark TS-MAPSCH remaps more VNRs than ReViNE. This is because TS-MAPSCH remaps as many VNRs as possible to obtain the maximum profit. In contrast, as defined in the reward function stated in (24), ReViNE aims to minimize the VNF deployment cost, which increases with the number of remapped VNFs. However, for high traffic volume, the VNF remapping rate for ReViNE increases slightly. In particular, for a VNR arrival rate of 0.2 services/s, the VNR remapping rate of ReViNE is 9.08%, which is 22.29% less than that of TS-MAPSCH. The reason is that the number of VNRs violating

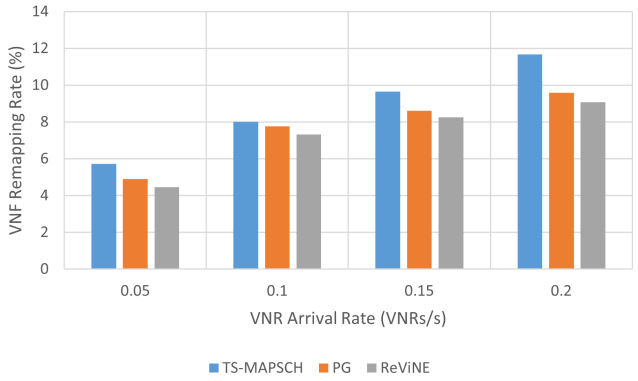

Fig. 6: Rate of VNF Remapping

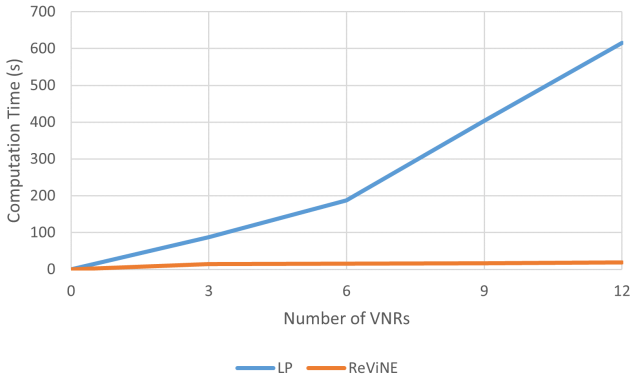

Fig. 8: Comparison of Computation Time with LP Solution

the delay bound increases significantly for high traffic volume. Therefore, the RL agent remaps more requests optimizing the VNF deployment cost and service revenue. From the simulation results, we can conclude that the proposed ReViNE scheme is more beneficial than the benchmarks as it offers a balanced rearrangement of an existing VNE strategy even for high traffic volume.

We also analyze the performance of the proposed RL model. Figure 7 depicts the average training time and average prediction time of the DQN for different VNR arrival rates. After successful training, the RL agent takes 1.69 ms on average to predict the VNE strategy. Hence, from the simulation results, we yield that the formulated RL model can obtain the optimal action online, a required feature for the scenario we described in Section I-A.

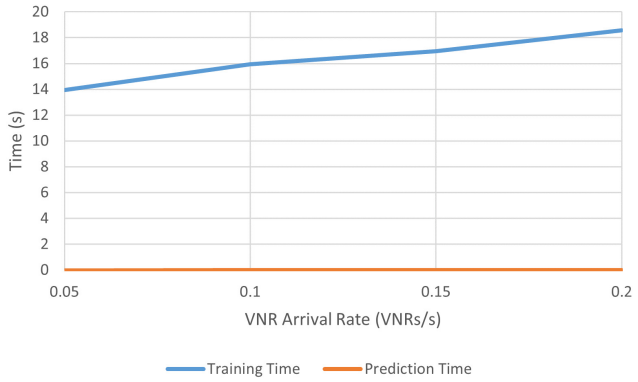

Fig. 7: Performance of RL Model

Figure 8 compares the computation time of ReViNE with the solution of LP formulated in (19). The LP is solved using the CVX solver [27]. The computation time of ReViNE includes both the training time and the prediction time. From the simulation results, we observe that the computation time of the LP solution increases exponentially with the number of VNRs. In addition, Figure 9 shows that ReViNE offers similar performance as the LP solution.

The simulation results demonstrate that ReViNE effectively mitigates the impact of topology changes, resulting in a higher acceptance ratio. This accomplishment is attributed

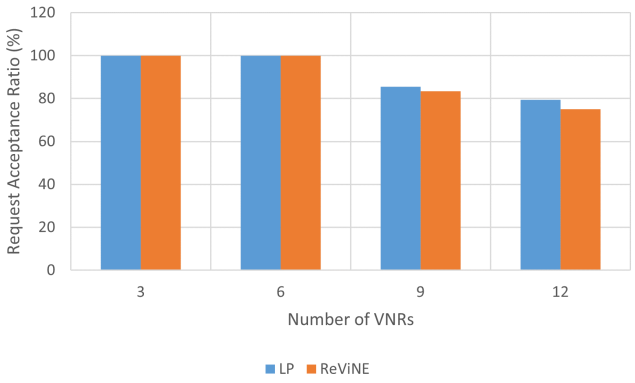

Fig. 9: Comparison of Acceptance Ratio with LP Solution

to ReViNE's strategy of mapping each VNR to a stable routing path. Moreover, ReViNE minimizes VNF redeployment, resulting in a lower remapping rate, especially at high traffic volumes where the RL agent optimizes cost and service revenue. Also, DQN achieves better performance than PG because DQN balances exploration and exploitation efficiently, leveraging experience replay for effective learning from past decisions.

VI. Conclusion

This paper presented an approach for RL-based dynamic VNE in STNs. The proposed scheme, ReViNE, determines appropriate satellite nodes and links for mapping the VNFs and virtual links of diverse VNRs considering link connectivity, link capacity, and processing capacity of the substrate STN. Additionally, ReViNE optimizes the VNR deployment cost and service revenue with selective remapping of the VNFs. We compared ReViNE with existing solutions. Simulation results exhibit that ReViNE increases the VNR acceptance ratio and the service revenue compared to the benchmarks optimizing the trade-off between VNF remapping and VNR deployment cost. Therefore, from the simulation results, we conclude that the dynamic VNE strategy generated by ReViNE is suitable for both the service provider and UE.

The current results motivate several future research topics. One potential topic is to consider multi-layer satellite

constellations involving a combination of LEO and medium earth orbit (MEO) satellites. In such a multi-orbit architecture, it is necessary to investigate the performance of the proposed scheme combined with VNF scheduling aspects. Another promising topic is the exploration of advanced reinforcement learning algorithms, which can provide the desired performance-complexity trade-off considering the large satellite topology.

REFERENCES

[1] G. Wang, S. Zhou, S. Zhang, Z. Niu, and X. Shen, "SFC-Based Service Provisioning for Reconfigurable Space-Air-Ground Integrated Networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 7, pp. 1478–1489, 2020.

[2] M. Minardi, Y. Drif, T. X. Vu, I. Maity, C. Politis, and S. Chatzinotas, "SDN-based Testbed for Emerging Use Cases in Beyond 5G NTN-Terrestrial Networks," in *IEEE/IFIP Network Operations and Management Symposium*, 2023, pp. 1–6.

[3] N. Jalodia, S. Henna, and A. Davy, "Deep Reinforcement Learning for Topology-Aware VNF Resource Prediction in NFV Environments," in *IEEE Conference on NFV-SDN*, 2019, pp. 1–5.

[4] M. Minardi, S. K. Sharma, S. Chatzinotas, and T. X. Vu, "A Parallel Link Mapping for Virtual Network Embedding with Joint Load-Balancing and Energy-Saving," in *IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 419–424.

[5] O. Alhussein, P. T. Do, Q. Ye, J. Li, W. Shi, W. Zhuang, X. Shen, X. Li, and J. Rao, "A Virtual Network Customization Framework for Multicast Services in NFV-Enabled Core Networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1025–1039, 2020.

[6] N. Kiran, X. Liu, S. Wang, and C. Yin, "VNF Placement and Resource Allocation in SDN/NFV-Enabled MEC Networks," in *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2020, pp. 1–6.

[7] F. Mendoza, M. Minardi, S. Chatzinotas, L. Lei, and T. X. Vu, "An SDN Based Testbed for Dynamic Network Slicing in Satellite-Terrestrial Networks," in *IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 36–41.

[8] L. Liu, S. Guo, G. Liu, and Y. Yang, "Joint Dynamical VNF Placement and SFC Routing in NFV-Enabled SDNs," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 4, pp. 4263–4276, Dec. 2021.

[9] J. Li, W. Shi, H. Wu, S. Zhang, and X. Shen, "Cost-Aware Dynamic SFC Mapping and Scheduling in SDN/NFV-Enabled Space-Air-Ground Integrated Networks for Internet of Vehicles," *IEEE Internet Things J.*, pp. 1–1, 2021, DOI: 10.1109/JIOT.2021.3058250.

[10] O. Kodheli, E. Lagunas, N. Maturo, S. K. Sharma, B. Shankar, J. F. M. Montoya, J. C. M. Duncan, D. Spano, S. Chatzinotas, S. Kisseleff, J. Querol, L. Lei, T. X. Vu, and G. Goussetis, "Satellite Communications in the New Space Era: A Survey and Future Challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 1, pp. 70–109, 2021.

[11] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. Mendoza Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani, E. Lagunas, and B. Ottersten, "Evolution of Non-Terrestrial Networks From 5G to 6G: A Survey," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2633–2672, 2022.

[12] D. M. Casas-Velasco, O. M. C. Rendon, and N. L. S. da Fonseca, "DRSIR: A Deep Reinforcement Learning Approach for Routing in Software-Defined Networking," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 4, pp. 4807–4820, 2022.

[13] M. U. Younus, M. K. Khan, and A. R. Bhatti, "Improving the Software-Defined Wireless Sensor Networks Routing Performance Using Reinforcement Learning," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3495–3508, 2022.

[14] I. Maity and T. Taleb, "ReSQ: Reinforcement Learning-Based Queue Allocation in Software-Defined Queuing Framework," *ournal of Networking and Network Applications*, vol. 2, no. 4, pp. 143–152, 2022, DOI: https://doi.org/10.33969/J-NaNA.2022.020402.

[15] P. M. Mohan and M. Gurusamy, "Resilient VNF Placement for Service Chain Embedding in Diversified 5G Network Slices," in *IEEE GLOBECOM*, 2019, pp. 1–6.

[16] Z. Tu, H. Zhou, K. Li, G. Li, and Q. Shen, "A Routing Optimization Method for Software-Defined SGIN Based on Deep Reinforcement Learning," in *IEEE Globecom Workshops*, 2019, pp. 1–6.

[17] S. Yuan, Y. Sun, and M. Peng, "Joint Network Function Placement and Routing Optimization in Dynamic Software-defined Satellite-Terrestrial Integrated Networks," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2023.

[18] H. Yang, W. Liu, X. Wang, and J. Li, "Group Sparse Space Information Network With Joint Virtual Network Function Deployment and Maximum Flow Routing Strategy," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5291–5305, 2023.

[19] Y. Yue, X. Tang, W. Yang, X. Zhang, Z. Zhang, C. Gao, and L. Xu, "Delay-aware and Resource-efficient VNF placement in 6G Non-Terrestrial Networks," in *IEEE WCNC*, 2023, pp. 1–6.

[20] M. Setayesh and V. W. Wong, "Service Function Chain Reconfiguration in 5G Core Networks Using Deep Learning," in *2021 IEEE Globecom*, 2021, pp. 1–6.

[21] J. S. Pujol Roig, D. M. Gutierrez-Estevez, and D. Gündüz, "Management and Orchestration of Virtual Network Functions via Deep Reinforcement Learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 304–317, 2020.

[22] M. Zhu, Q. Chen, J. Gu, and P. Gu, "Deep Reinforcement Learning for Provisioning Virtualized Network Function in Inter-Datacenter Elastic Optical Networks," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 3, pp. 3341–3351, 2022.

[23] Z. Song, Y. Hao, Y. Liu, and X. Sun, "Energy-Efficient Multiaccess Edge Computing for Terrestrial-Satellite Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14 202–14 218, 2021.

[24] T. Kim, J. Kwak, and J. P. Choi, "Satellite Edge Computing Architecture and Network Slice Scheduling for IoT Support," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14 938–14 951, 2022.

[25] P. Wang, X. Zhang, S. Zhang, H. Li, and T. Zhang, "Time-Expanded Graph-Based Resource Allocation Over the Satellite Networks," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 360–363, 2019.

[26] M. Minardi, T. X. Vu, L. Lei, C. Politis, and S. Chatzinotas, "Virtual Network Embedding for NGSO Systems: Algorithmic Solution and SDN-Testbed Validation," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[27] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[28] F. Z. Mardi, M. Bagaa, Y. Hadjadj-Aoul, and N. Benamar, "Heuristic-Deep Q-Network-Based Network Slicing in LoRaWAN," in *IEEE ICC*, 2023, pp. 4731–4736.

[29] P. Tehrani, F. Restuccia, and M. Levorato, "Federated Deep Reinforcement Learning for the Distributed Control of NextG Wireless Networks," in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 248–253.

[30] I. Maity, T. X. Vu, and S. Chatzinotas, "D-Schedule: Dependency-Aware VNF Scheduling in Satellite-Terrestrial Networks," in *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2023, pp. 1283–1288.

[31] Z. Li, Y. Kong, and C. Jiang, "A Transfer Double Deep Q Network Based DDoS Detection Method for Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 72, no. 4, pp. 5317–5331, 2023.

[32] "CELESTRAK," Accessed: Jun. 2023. [Online]. Available: https://celestrak.org

[33] 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.*, pp. 1–11, 2021.

[34] X. Qin, T. Ma, Z. Tang, X. Zhang, H. Zhou, and L. Zhao, "Service-Aware Resource Orchestration in Ultra-Dense LEO Satellite-Terrestrial Integrated 6G: A Service Function Chain Approach," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2023.

[35] X. Gao, R. Liu, and A. Kaushik, "Virtual Network Function Placement in Satellite Edge Computing With a Potential Game Approach," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 2, pp. 1243–1259, 2022.

[36] "MATLAB Satellite Communications Toolbox," Accessed: Jun. 2023. [Online]. Available: https://nl.mathworks.com/products/satellite-communications.html

[37] J. Guo, D. Rincón, S. Sallent, L. Yang, X. Chen, and X. Chen, "Gateway Placement Optimization in LEO Satellite Networks Based on Traffic Estimation," *IEEE Trans. Veh. Technol.*, vol. 70, no. 4, pp. 3860–3876, 2021.

Ilora Maity (Member, IEEE) received the B.Tech., the M.E., and the Ph.D. all in Computer Science and Engineering, from West Bengal University of Technology, India, the Indian Institute of Engineering Science and Technology, Shibpur, India, and the Indian Institute of Technology Kharagpur, India in 2008, 2011, and 2021 respectively. She has worked as a Technical Analyst of Banking & Financial Services in Cognizant Technology Solutions India Pvt. Ltd., Kolkata, India, from September 2011 to July 2015. From January 2021 to April 2021, she was a Postdoctoral Researcher at Aalto University, Finland. She is a Research Associate at the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg. Her research interests include Software-Defined Networking, Network Function Virtualization, the Internet of Things, satellite communications, and quantum communications. She has authored over 30 technical papers in refereed international journals and conferences. Dr. Maity also served as a TPC member for several IEEE conferences. She is on the editorial board of SciencePG International Journal of Wireless Communications and Mobile Computing.

Thang X. Vu (Senior Member, IEEE) received the B.S. and the M.Sc., both in Electronics and Telecommunications Engineering, from the VNU University of Engineering and Technology, Vietnam, in 2007 and 2009, respectively, and the Ph.D. in Electrical Engineering from the University Paris-Sud, France, in 2014. In 2010, he received the Allocation de Recherche fellowship to study Ph.D. in France. From July 2014 to January 2016, he was a postdoctoral researcher with the Singapore University of Technology and Design (SUTD), Singapore. Currently, he is a research scientist at the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg. His research interest includes wireless communications, with particular interests of applications of optimization and machine learning on design and analyze the multi-layer 6G networks. He has successfully acquired, as the PI and vice PI, several Luxembourg national and ESA projects with a total funding of 2.6 MEURs. He was a recipient of the SigTelCom 2019 Best Paper Award. He is also serving as an Associate Editor for the IEEE Communications Letters.

Symeon Chatzinotas (S'06, M'09, SM'13, F'23) is Full Professor and Head of the SIGCOM Research Group at SnT, University of Luxembourg. He is coordinating the research activities on communications and networking across a group of 80 researchers, acting as a PI for more than 40 projects and main representative for 3GPP, ETSI, DVB. He is currently serving in the editorial board of the IEEE Transactions on Communications, IEEE Open Journal of Vehicular Technology and the International Journal of Satellite Communications and Networking. In the past, he has been a Visiting Professor at the University of Parma, Italy and was involved in numerous RD projects for NCSR Demokritos, CERTH Hellas and CCSR, University of Surrey. He was the co-recipient of the 2014 IEEE Distinguished Contributions to Satellite Communications Award and Best Paper Awards at WCNC, 5GWF, EURASIP JWCN, CROWNCOM, ICSSC. He has (co-)authored more than 700 technical papers in refereed international journals, conferences and scientific books.