Admission Control and Embedding of Network Slices with Flexible VNF Order

Quang-Trung Luu*§, Minh-Thanh Nguyen*, Tai-Hung Nguyen*, Michel Kieffer†,
Van-Dinh Nguyen‡, Quang-Lap Luu*, and Trung-Toan Nguyen*
*School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, Hanoi 100000, Vietnam
†Université Paris-Saclay - CNRS - CentraleSupélec - L2S, Gif-sur-Yvette, F-91192, France
‡College of Engineering and Computer Science & Center for Environmental Intelligence,
VinUniversity, Vinhomes Ocean Park, Hanoi 100000, Vietnam
§Corresponding author. E-mail: trung.luuquang@hust.edu.vn

***Abstract*—Network slicing has appeared a key feature in 5G and beyond communication networks that enables the creation of multiple virtual networks (*i.e.*, *slices*) over a shared physical network infrastructure. This process involves efficiently embedding (or mapping) each slice element, including virtual network functions (VNFs) and their interconnections, onto the physical network. This paper explores a scenario where the order of VNFs can be adjusted during slice embedding, offering greater flexibility to increase the number of services deployed on the infrastructure. We formulate a novel optimization framework to tackle the challenges of slice admission control and embedding with this flexibility. A heuristic is also introduced to derive embedding solutions in a timely manner. Simulation results demonstrate that allowing flexible VNF ordering significantly increases the number of slices that can be deployed in the network infrastructure.**

***Index Terms*—Network slicing, admission control, slice embedding, resource allocation, 5G and beyond, flexible order, open radio access network, integer linear programming.**

I. Introduction

Network slicing is a transformative technology in 5G and beyond communication systems that allows the creation of multiple virtual networks on a shared physical infrastructure [1]. Each virtual network (called *slice*) is tailored to meet the specific needs and requirements of different applications and services, ranging from high-bandwidth video streaming to low-latency industrial control systems. This customization is achieved by allocating dedicated network resources and configuring them to provide the necessary performance characteristics, such as bandwidth, latency, and reliability [2].

One of the critical challenges in the realm of network slicing is the problem of network slice embedding (NSE) (also referred to as service function chain embedding or virtual network embedding). This involves the task of mapping virtual network functions (VNFs) and their interconnections (called *virtual links*) onto the physical network infrastructure in an optimized manner. The process of embedding these VNFs onto the physical network requires careful consideration of various factors, including resource availability, network topology, and service level agreements (SLAs). An optimal embedding ensures that the network resources are utilized efficiently, the performance requirements of each slice are met, and the operational costs are minimized. Nevertheless, achieving this optimal embedding is a complex and computationally challenging problem, especially in large-scale and dynamic network environments [1], [2].

Most prior works on slicing considered slices with fixed structures, *i.e.*, its VNFs are chained with fixed position in advance [3], [4], [5], [6], [7], [8]. Nevertheless, in practical, several variants of slice structures may be considered to implement the same service. These variants may imply different orders of some VNFs, or even different types of VNFs forming the slices [9], [10]. For instance, there is no strict order between a proxy server and a WAN optimizer. [9]. Consequently, these VNFs can flexibly be placed to compose the slice. On the other hand, some VNFs might have to be fixed in specific positions, *e.g.*, the virtual distributed unit (DU) and the virtual central unit (CU) have to be placed right after a radio unit (RU) in a typical open radio access network (Open RAN) [11]. Additionally, different resource requirements and quality of service may be associated to each of these variants. The possibility to choose among several variants of slices to implement a service provides additional flexibility to the virtual network operator (VNO) during slice embedding.

A few works address the flexibility of VNF order and the composition of network slices can be found in the literature. For instance, the work in [9] proposed an integer linear programming (ILP) approach to optimally solve the problem of service function chain composition by characterizing service requests in terms of VNFs and determining the optimal chain.

In [12], a YANG data model was proposed to support the flexibility in ordering VNFs within a network service, *i.e.*, the order of some VNFs can be swapped without affecting the overall functionality of the slice. In the proposed system, the network orchestration system can select the ideal combination of service components to achieve the most effective service placement within the network. Nevertheless, this work did not provide an optimization framework to address the problem of network slice embedding with flexible VNF order.

Contributions. This paper aims at formalizing the slice embedding problem when flexibility in the order of some VNFs in the slices is allowed. Compared to classical network slice embedding, in the proposed approach, the structure of the

978-3-903176-66-9 ©2024 IFIP

slice has to be jointly optimized with the embedding. As a result, a nonlinear integer programming problem is obtained. To tackle this problem, we propose some linearization techniques to obtain an ILP. In this paper, we show that allowing flexibility in selecting the VNF order leads to an increase of the slice acceptance rate, *i.e.*, more slices can be deployed on a constrained physical network.

The remainder of this paper is organized as follows. Sec. II introduces the flexible slice embedding problem. A heuristic approach is introduced in Sec. III, followed by the simulation results in Sec. IV. Finally, Sec. V concludes this work and gives some perspectives.

II. Problem Statement

In this section, we formalize the problem of admission control and embedding of slices that allow flexibility in the order of their VNFs. Upon receiving the request for deploying a given slice, the VNO can then choose the order of VNFs to form the slice that provides the best embedding solution, *e.g.*, in terms of deployment cost.

A. Network Model

The physical network infrastructure is represented as a weighted directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of physical nodes and $\mathcal{L}$ is the set of all available links within $\mathcal{G}$. Each physical node $i \in \mathcal{N}$ is characterized by its available resource capacity A_i (*e.g.*, computing, storage, or processing). Similarity, each physical link $ij \in \mathcal{L}$ has an available resource capacity A_{ij} (*e.g.*, bandwidth).

B. Slice Model

For each slice $s \in \mathcal{S}$, let $\mathcal{N}_s$ be the set of virtual nodes (representing the VNFs). Each virtual node $v \in \mathcal{N}_s$ requires an amount R_v of physical resources. The set $\mathcal{P}_s = \{(v, w), v \in \mathcal{N}_s, w \in \mathcal{N}_s : w \neq v\}$ represents all possible combinations of VNF pairs which may be connected by a virtual link.

To simplify presentation, we consider slices represented by linear chains, without loops or branches. We assume, moreover, that some virtual nodes $v \in \mathcal{N}_s$ have a fixed position, while others may be placed flexibly in the slice. In what follows, a slice *configuration* represents a possible organization of the VNFs within that slice.

Fig. 1 illustrates a slice with flexible VNF order, adapted from the example slice in [13]. This slice is dedicated to video streaming service at downlink and has five VNFs, including an intrusion detection and prevention system (IDPS), a video optimization controller (VOC), a traffic monitoring (TM), a gateway (GW), and a DU. The IDPS, GW, and DU functions are placed at fixed positions 1, 2, and 5, respectively. The other VNFs, VOC and TM, can be flexibly placed at positions 3 or 4, leading to two possible slice configurations, (IDPS $\rightarrow$ VOC $\rightarrow$ TM $\rightarrow$ GW $\rightarrow$ DU) and (IDPS $\rightarrow$ TM $\rightarrow$ VOC $\rightarrow$ GW $\rightarrow$ DU).

In general, different slice configurations may lead to different resource requirements of the VNFs and of the virtual links between VNFs. For the sake of simplicity, we consider that all possible slice configurations share the same VNF resource requirements. They only differ by the resource requirements of the virtual links.

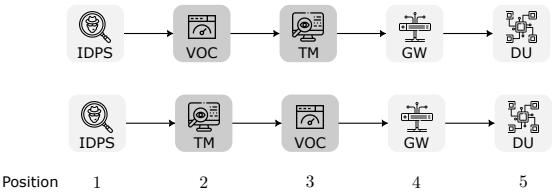


Fig. 1. An example of slice with flexible VNF order. The functions TM and VOC can be swapped to the other, creating two possible slice configurations.

C. Variables

Upon receiving the request for slice s, the VNO has to determine the slice combination to use so as to provide the requested service. One introduces the set of binary variables $\boldsymbol{y} = \{y^{vw,s}\}_{s\in\mathcal{S}, vw\in\mathcal{P}_s}$, where $y^{vw,s}$ to indicate whether a link combination vw is chosen to form slice s, *i.e.*, $y^{vw,s} = 1$ if virtual link vw is chosen, and $y^{vw,s} = 0$, otherwise. The set of all virtual links[1] used in an slice configuration can be represented by $\mathcal{L}_s = \{vw \in \mathcal{P}_s : y^{vw,s} = 1\}$. One also introduces the variable set $\boldsymbol{\theta} = \{\theta_p^{v,s}\}_{s\in\mathcal{S}, v\in\mathcal{N}_s, p\in[1,|\mathcal{N}_s|]}$, where $\theta_p^{v,s}$ represents the position of each VNF in slice s, $\theta_p^{v,s} = 1$ if v is placed at the p^{th} position of slice s, and $\theta_p^{v,s} = 0$, otherwise. The position of VNF v in slice s is then $p^{v,s} = \sum_{p=1}^{|\mathcal{N}_s|} p\theta_p^{v,s}, \forall v \in \mathcal{N}_s$.

Now, two sets of variables are introduced to formalize the slice embedding problem, namely $\boldsymbol{\pi} = \{\pi_s\}_{s\in\mathcal{S}}$ and $\boldsymbol{x} = \{x_i^{v,s}, x_{ij}^{vw,s}\}_{(i,ij)\in\mathcal{G},(v,vw)\in\mathcal{G}_s,s\in\mathcal{S}}$. Here, the variable $\pi_s \in \{0, 1\}$ indicates whether a slice is accepted ($\pi_s = 1$) or rejected ($\pi_s = 0$). The node mapping indicator variables $x_i^{v,s} \in \{0, 1\}$ indicates whether VNF v of slice s is mapped onto the physical node $i \in \mathcal{N}$ ($x_i^{v,s} = 1$) or not ($x_i^{v,s} = 0$). Similarly for the link mapping indicator $x_{ij}^{vw,s} \in \{0, 1\}$ between virtual link vw of slice s and physical link ij.

D. Constraints

Node capacity. The total requested resources in a physical node must never exceed its available resources

$$\sum_{s\in\mathcal{S}} \sum_{v\in\mathcal{N}_s} x_i^{v,s} R_v \leq A_i, \qquad \forall i \in \mathcal{N}. \tag{1}$$

Link capacity. The total requested resources in a physical link should never exceed its available resources,

$$\sum_{s\in\mathcal{S}} \sum_{v,w\in\mathcal{N}_s} x_{ij}^{vw,s} y^{vw,s} R_{vw} \leq A_{ij}, \quad \forall ij \in \mathcal{L}. \tag{2}$$

The quadratic term $x_{ij}^{vw,s} y^{vw,s}$ makes (2) nonlinear. To overcome this issue, consider the additional variable $z_{ij}^{vw,s} \in \{0, 1\}$ and the following constraints to linearize (2),

$$\sum_{s\in\mathcal{S}} \sum_{v,w\in\mathcal{N}_s} z_{ij}^{vw,s} R_{vw} \leq A_{ij}, \tag{3a}$$

[1]It is worth noting that, $\mathcal{L}_s$ is not known in advance. This set is only determined after a successful mapping of slice s onto the physical network.

$$z_{ij}^{vw,s} \leq x_{ij}^{vw,s} \text{ and } z_{ij}^{vw,s} \leq y^{vw,s}, \tag{3b}$$
$$z_{ij}^{vw,s} \geq x_{ij}^{vw,s} + y^{vw,s} - 1, \tag{3c}$$
$$\forall ij \in \mathcal{E}, s \in \mathcal{S}, (v,w) \in \mathcal{P}_s. \tag{3d}$$

Mapped only once. As in [3], [14], we consider the following constraint to enforce that each physical node can host at most one VNF of a given slice. This is to ensure that, in case of node failure, only a single VNF has to be relocated.

$$\sum_{v \in \mathcal{N}_s} x_i^{v,s} \leq \pi_s \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}. \tag{4}$$

Accepted slices are served. This constraint guarantees that, once slice s is accepted (*i.e.*, $\pi_s = 1$), all of its VNFs should be mapped onto the physical network,

$$\sum_{i \in \mathcal{N}} x_i^{v,s} = \pi_s \qquad \forall v \in \mathcal{N}_s, s \in \mathcal{S} \tag{5}$$

Flow conservation. This constraint enforces that, once a virtual link vw belong to the chosen configuration for slice s, *i.e.*, $y^{vw,s} = 1$, the mapping of vw onto one or several physical links should preserve the traffic between VNFs v and w,

$$\sum_{j \in \mathcal{N}} x_{ij}^{vw,s} - \sum_{j \in \mathcal{N}} x_{ji}^{vw,s} = x_i^{v,s} - x_i^{w,s},$$
$$\forall i \in \mathcal{N}, s \in \mathcal{S}, (v,w) \in \mathcal{P}_s : y^{vw,s} = 1. \tag{6}$$

Constraint (6) has only to be active when $y^{vw,s} = 1$. To address this issue, the big-M method is considered as follows

$$\sum_{j \in \mathcal{N}} x_{ij}^{vw,s} - \sum_{j \in \mathcal{N}} x_{ji}^{vw,s} - x_i^{v,s} + x_i^{w,s} \leq M\left(1 - y^{vw,s}\right),$$
$$\sum_{j \in \mathcal{N}} x_{ij}^{vw,s} - \sum_{j \in \mathcal{N}} x_{ji}^{vw,s} - x_i^{v,s} + x_i^{w,s} \geq -M\left(1 - y^{vw,s}\right),$$
$$\forall i \in \mathcal{N}, s \in \mathcal{S}, (v,w) \in \mathcal{P}_s. \tag{7}$$

Formation of virtual links. As only linear configurations are considered, once a given VNF pair (v,w) is chosen to form the virtual link vw, the following constraints have to be satisfied

$$\sum_{(v,w) \in \mathcal{P}_s} y^{vw,s} \leq \pi_s, \quad \forall s \in \mathcal{S}, v \in \mathcal{N}_s, \tag{8a}$$
$$\sum_{(v,w) \in \mathcal{P}_s} y^{vw,s} \leq \pi_s, \quad \forall s \in \mathcal{S}, w \in \mathcal{N}_s, \tag{8b}$$
$$\sum_{(v,w) \in \mathcal{P}'_s} y^{vw,s} \leq \pi_s\left(|\mathcal{N}'_s| - 1\right), \quad \forall \mathcal{N}'_s \subseteq \mathcal{N}_s. \tag{8c}$$

Position constraint. Once a virtual link vw is formed, *i.e.*, $y^{vw,s} = 1$, the position of v and w has to satisfy $p^{w,s} - p^{v,s} = 1$. One has thus

$$p^{w,s} - p^{v,s} = 1, \quad \forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s : y^{vw,s} = 1. \tag{9}$$

Similar to (6), (9) is nonlinear due to the condition $y^{vw,s} = 1$. We reuse the big-M method to linearize (9) as

$$1 - M\left(1 - y^{vw,s}\right) \leq p^{w,s} - p^{v,s} \leq 1 + M\left(1 - y^{vw,s}\right),$$
$$\forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s. \tag{10}$$

Remove redundant $x_{ij}^{vw,s}$. To ensure that, if $y^{vw,s} = 0$, *i.e.*, no virtual link is formed between VNFs v and w, the following constraint is introduced

$$\sum_{ij \in \mathcal{L}} x_{ij}^{vw,s} = 0, \quad \forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s : y^{vw,s} = 0. \tag{11}$$

Again, as in (6) and (9), the big-M method is used to linearize (11) as follows,

$$\sum_{ij \in \mathcal{L}} x_{ij}^{vw,s} \leq M y^{vw,s}, \quad \forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s, \tag{12a}$$
$$\sum_{ij \in \mathcal{L}} x_{ij}^{vw,s} \geq -M y^{vw,s}, \quad \forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s. \tag{12b}$$

Only one position. The following constraints guarantee that, once a given slice s is admitted, each VNF v of slice s should occupy only one position p in the chain (13a) and each position p is occupied by only one VNF (13b),

$$\sum_{p=1}^{|\mathcal{N}_s|} \theta_p^{v,s} = \pi_s, \quad \forall s \in \mathcal{S}, v \in \mathcal{N}_s, \tag{13a}$$
$$\sum_{v \in \mathcal{N}_s} \theta_p^{v,s} = \pi_s, \quad \forall s \in \mathcal{S}, p \in [1, |\mathcal{N}_s|]. \tag{13b}$$

Fixed order and position constraints. In practice, some VNFs may have a fixed position in the slice, *e.g.*, in RAN slicing, the virtual distributed unit (vDU) should be placed at the beginning (for the uplink) or the end (for the downlink) of the slice [11]. Similarly, the order of some VNFs may have to follow some strict rules, *e.g.*, in a video streaming slice, the virtual firewall is placed before the virtual traffic monitor [15]. This constrains the positions of some VNFs, and also of some virtual links. Having this, we denote by $\mathcal{P}_s^{\mathrm{F}} \subset \mathcal{P}_s$ the set of VNF pairs that have been *fixed* to form virtual links and by $\mathcal{N}_s^{\mathrm{F}} \subset \mathcal{N}_s$ the set of VNFs whose positions are *fixed* in the slice, *i.e.*, $\mathcal{P}_s^{\mathrm{F}} = \{(v,w) \in \mathcal{P}_s : y^{vw,s} = 1\}$ and $\mathcal{N}_s^{\mathrm{F}} = \{v \in \mathcal{N}_s : \theta_p^{v,s} = 1\}$. One has thus

$$y^{vw,s} = 1, \quad \forall s \in \mathcal{S}, (v,w) \in \mathcal{P}_s^{\mathrm{F}}, \tag{14a}$$
$$\theta_p^{v,s} = \pi_s, \quad \forall s \in \mathcal{S}, v \in \mathcal{N}_s^{\mathrm{F}}. \tag{14b}$$

Finally, the problem of admission control and embedding of slices with flexible ordered VNFs (denoted as SE-FlexOrder), which aims to maximize the number of accepted slices and minimizing the number of used physical links is described as

$$\max_{\boldsymbol{\pi}, \boldsymbol{x}} \quad \gamma N(\boldsymbol{\pi}) - (1-\gamma) \sum_{s \in \mathcal{S}} H_s \qquad \text{(SE-FlexOrder)}$$
$$\text{s.t.} \quad (1), (3), (4), (5), (7), (8), (10), (12), (13), (14),$$

where $N(\boldsymbol{\pi}) = \sum_{s \in \mathcal{S}} \pi_s$ is the number of accepted slices and $H_s = \sum_{vw \in \mathcal{L}_s} \sum_{ij \in \mathcal{L}} x_{ij}^{vw,s}$ represents the number of physical links used by slice s and $\gamma \in [0,1]$ is a tuning parameter to balance the value between the first and the second term of the objective function of Problem (SE-FlexOrder).

When the VNF order flexibility is allowed, there exists a polynomial time reduction of the problem of embedding network slices with fixed VNF order [3], [5] (denoted as SE-Fixed) to Problem (SE-FlexOrder). Since Problem (SE-Fixed) is *NP*-hard, Problem (SE-FlexOrder) is also *NP*-hard.

III. Heuristic Approach

To demonstrate the benefit of allowing VNF ordering flexibility when using ILP-SE, we also deploy a simple heuristic, namely MRN-SE (Most Resource Neighbor Slice Embedding), which provides low complexity and improved scability and near-optimal solutions in a timely manner. Briefly, MRN-SE tries to map each possible configuration of each slice onto the physical network using a greedy approach, *i.e.*, selecting the physical node providing the most resource; and the Dijkstra's algorithm to find physical links to map the virtual links between the already mapped VNFs. Then, MRN-SE selects the configuration that provides the best objective value to Problem (SE-FlexOrder). If multiple configurations provide the same best objective value, a random one is chosen.

IV. Performance Evaluation

A. Simulation setup

Physical network. The physical network is generated from a fat-tree topology, as in [3], [4]. This topology consists of four layers: Core, Aggregation, Edge, and Host. A binary fat-tree has 2 core nodes, 4 aggregation nodes, 4 edge nodes, and 8 host nodes. Each physical node provides compute power as the number of available vCPUs and storage space. Details of the network setup are shown in Fig. 2.

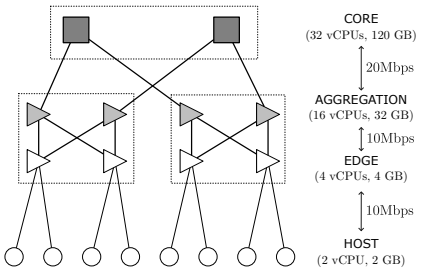
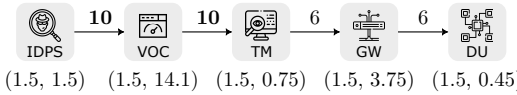


Fig. 2. The physical network used for simulation. Each node provides a fixed amount of compute and storage resources, measured in the number of vCPUs and GBs respectively. Links have a fixed amount of available bandwidth, measured in Mbps.

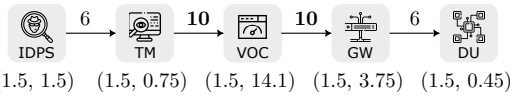
Slices. We reconsider the example slice shown in Fig. 1 with two possible configurations. Each slice is designed to provide a high-definition video streaming service to multiple end-devices. Resource requirements of each configuration are adopted from [13] and detailed in Fig. 3.

Simulation scenarios. We consider two scenarios with different number of slices: $|\mathcal{S}| = 15$ and $|\mathcal{S}| = 20$ slices. To investigate the benefit of VNF ordering flexibility, we compare the performance of ILP-SE and MRN-SE in three different settings: (i) when only the first slice configuration in Figure 3 is chosen ("k_1-only"); (ii) when only the second configuration is chosen ("k_2-only"); and (iii) when a flexibility of choosing between two configurations is allowed for the slice embedding ("flexible").

In addition, the tuning parameter γ in the objective function of Problem (SE-FlexOrder) is set to 0.999, since the value of H_s (number of used physical links) is typically much larger than that of $N(\boldsymbol{\pi})$ (number of accepted slices).



(a) Configuration 1 (k_1).



(b) Configuration 2 (k_2).

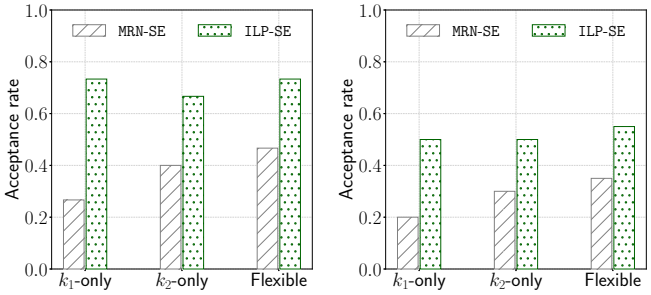
Fig. 3. The two network slice configurations used for simulation. Each node requires an amount of compute and storage resources (measured in terms of number of vCPUs and GBs, respectively). Virtual links between VNFs require a certain amount of bandwidth (measured in Gbps).

Software and hardware. All simulations are performed on a PC with Intel i5-3320M CPU and 16 GB of RAM. Both ILP-SE and MRN-SE are written using Python 3.10.14. ILP-SE uses Gurobi Optimizer v11.0.1 as the ILP solver.

Metrics. Two metrics are used to perform the evaluation: (i) the total number of accepted slices (*i.e.*, $N(\boldsymbol{\pi})$) and (ii) the total time used to compute the results.

B. Results

Fig. 4 shows the slice acceptance rate of ILP-SE and MRN-SE in three different settings: k_1-only, k_2-only, and flexible between two configurations in two test cases ($|\mathcal{S}| = 15$ and $|\mathcal{S}| = 20$). Fig. 5 shows the acceptance rate per configuration that each approach obtains in the "flexible" setting.

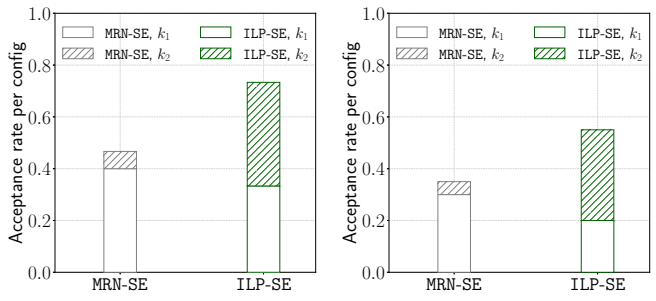


(a) $|\mathcal{S}| = 15$. (b) $|\mathcal{S}| = 20$.

Fig. 4. Slice acceptance rate of ILP-SE and MRN-SE when (a) $|\mathcal{S}| = 15$ and (b) $|\mathcal{S}| = 20$.

As shown in Fig. 4a and Fig. 4b, both algorithms succeed in mapping more network slices onto the physical network infrastructure in the flexible setting, compared to the two settings k_1-only and k_2-only when only one configuration is selected. Precisely, when $|\mathcal{S}| = 20$, ILP-SE yields a performance gap of 10% when performing a flexible mapping, compared to that of the k_1-only and k_2-only (see Fig. 4b). This confirms the advantage of allowing flexibility in choosing the order of VNFs leading to a better slice acceptance rate. In addition, as expected, ILP-SE outperforms MRN-SE in terms of acceptance rate, as shown in the same figure.

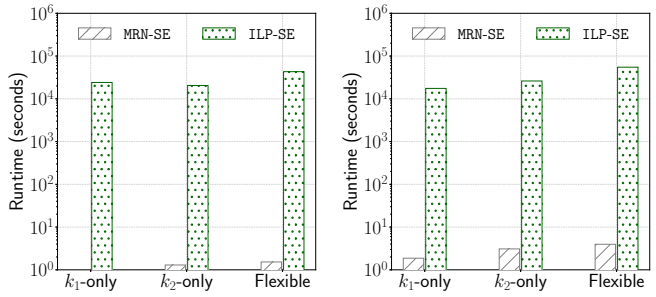
Fig. 5 shows that MRN-SE favors the use of configuration k_1, as it has cascade bandwidth requirements (see Fig. 3a) similar



(a) $|\mathcal{S}| = 15$. (b) $|\mathcal{S}| = 20$.

Fig. 5. Slice acceptance rate per configuration of ILP-SE and MRN-SE when (a) $|\mathcal{S}| = 15$ and (b) $|\mathcal{S}| = 20$.

to the available bandwidth in the fat-tree network. On the other hand, ILP-SE is able to balance the use of two configurations, leading to a better slice embedding performance.



(a) $|\mathcal{S}| = 15$. (b) $|\mathcal{S}| = 20$.

Fig. 6. Execution time (in seconds) of ILP-SE and MRN-SE when (a) $|\mathcal{S}| = 15$ and (b) $|\mathcal{S}| = 20$.

Fig. 6 illustrates the total time required to compute the embedding results of two approaches in the three different settings of the two test cases. For the last setting, the time to select the best slice configuration is also considered. It can be observed that the time required by ILP-SE to yield slice embedding solutions is much higher than that of MRN-SE, due to its *NP*-hardness. In addition, since introducing the flexibility adds more complexity to the problem, the compute time gets higher from k_1-only, k_2-only to flexible setting, for both algorithms. Nevertheless, it can be seen that MRN-SE is still able to yield embedding solutions in a reasonable time, as shown in Fig. 6b.

V. Conclusion

In this paper, we proposed an optimization framework to manage slice admission control and embedding with flexibility in the order of VNFs within slices. An integer linear program is obtained and an heuristic solution approach, namely MRN-SE, is proposed. It yields near-optimal solutions in a reasonable time, making it more scalable for large network settings. Simulation results demonstrate that exploiting the flexibility in VNF ordering can significantly increase the slice acceptance rate. These findings underscore the potential of flexible VNF ordering to maximize resource utilization and service deployment, paving the way for more dynamic and efficient future network infrastructures.

For future work, we plan to investigate this problem in more complex scenarios, with different network slice configurations and physical network topologies. We may also consider using various deep reinforcement learning architectures to solve Problem (SE-FlexOrder).

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.04-2023.17.

References

[1] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, 2017.

[2] R. Su, D. Zhang, R. Venkatesan, Z. Gong, C. Li, F. Ding, F. Jiang, and Z. Zhu, "Resource allocation for network slicing in 5G telecommunication networks: A survey of principles and models," *IEEE Netw.*, vol. 33, no. 6, pp. 172–179, 2019.

[3] R. Riggio, A. Bradai, D. Harutyunyan, T. Rasheed, and T. Ahmed, "Scheduling wireless virtual networks functions," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 2, pp. 240–252, 2016.

[4] N. Bouten, R. Mijumbi, J. Serrat, J. Famaey, S. Latré, and F. De Turck, "Semantically enhanced mapping algorithm for affinity-constrained service function chain requests," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, no. 2, pp. 317–331, 2017.

[5] P. Vizarreta, M. Condoluci, C. M. Machuca, T. Mahmoodi, and W. Kellerer, "QoS-driven function placement reducing expenditures in NFV deployments," in *Proc. 2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–7.

[6] Q.-T. Luu, S. Kerboeuf, and M. Kieffer, "Admission control and resource reservation for prioritized slice requests with guaranteed SLA under uncertainties," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 3, pp. 3136–3153, 2022.

[7] P. Zhang, N. Chen, S. Li, K.-K. R. Choo, C. Jiang, and S. Wu, "Multi-domain virtual network embedding algorithm based on horizontal federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 3363–3375, 2023.

[8] M.-T. Nguyen, Q.-T. Luu, T.-H. Nguyen, D.-M. Tran, T.-A. Do, K.-H. Do, and V.-H. Nguyen, "Accelerating network slice embedding with reinforcement learning," in *Proc. IEEE International Conference on Communications and Electronics (ICCE)*, 2024.

[9] A. F. Ocampo, J. Gil-Herrera, P. H. Isolani, M. C. Neves, J. F. Botero, S. Latré, L. Zambenedetti, M. P. Barcellos, and L. P. Gaspary, "Optimal service function chain composition in network functions virtualization," in *Proc. 11th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security (AIMS)*, 2017, pp. 62–76.

[10] J. Huang, Q. Duan, S. Guo, Y. Yan, and S. Yu, "Converged network-cloud service composition with end-to-end performance guarantee," *IEEE Trans. Cloud Comput.*, vol. 6, no. 2, pp. 545–557, 2015.

[11] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges," *IEEE Commun. Surv. Tutor.*, vol. 25, no. 2, pp. 1376–1411, 2023.

[12] S. Mehraghdam and H. Karl, "Placement of services with flexible structures specified by a YANG data model," in *Proc. 2016 IEEE International Conference on Network Softwarization (NetSoft)*, 2016, pp. 184–192.

[13] Q.-T. Luu, S. Kerboeuf, A. Mouradian, and M. Kieffer, "A coverage-aware resource provisioning method for network slicing," *IEEE/ACM Trans. Netw.*, vol. 28, no. 6, pp. 2393–2406, 2020.

[14] Q.-T. Luu, M. Kieffer, A. Mouradian, and S. Kerboeuf, "Aggregated resource provisioning for network slices," in *Proc. 2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[15] M. Savi, M. Tornatore, and G. Verticale, "Impact of processing-resource sharing on the placement of chained virtual network functions," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1479–1492, 2021.