Cost-Aware Dynamic SFC Mapping and Scheduling in SDN/NFV-Enabled Space–Air–Ground-Integrated Networks for Internet of Vehicles

Junling Li, *Graduate Student Member, IEEE*, Weisen Shi, *Graduate Student Member, IEEE*, Huaqing Wu, *Student Member, IEEE*, Shan Zhang, *Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Space–air–ground-integrated networks (SAGINs) are deemed as a promising solution to support multifarious Internet of Vehicles (IoV) services with diversified Quality-of-Service (QoS) requirements in future communication networks. Network function virtualization (NFV) and software-defined networking (SDN) are two complementary and promising technologies to reduce the function provisioning cost and coordinate the heterogeneous physical resources in SAGIN. In this article, we investigate the online dynamic virtual network function (VNF) mapping and scheduling in SAGIN, considering the dynamicity of IoV services. The VNF live migration, VNF reinstantiation, and VNF rescheduling are enabled to increase the service acceptance ratio and service provider's profits. Considering the heterogeneity of space, air, and ground nodes, we first model the migration cost and additional delay incurred by VNF live migration and reinstantiation. We then formulate the dynamic VNF mapping and scheduling jointly as a mixed-integer linear programming (MILP) problem with specified cost and delay models. We propose two Tabu search (TS)-based algorithms, i.e., TS-based VNF remapping and rescheduling (TS-MAPSCH) algorithm and TS-based pure VNF rescheduling (TS-PSCH) algorithm, to obtain suboptimal solutions to the MILP problem efficiently. Simulation results show that the proposed solution is very close to the optimum and that the proposed dynamic algorithms outperform existing works with respect to multiple performance metrics, including the service provider's profit, service acceptance ratio, and QoS satisfaction level.**

***Index Terms*—Internet of Vehicles (IoV), network function virtualization (NFV), resource allocation, software-defined networking (SDN), space–air–ground-integrated networks (SAGINs), virtual network function (VNF) mapping, VNF scheduling.**

I. Introduction

THE Internet of Vehicles (IoV) has been envisioned as a transformative technology to enhance driving safety, facilitate efficient traffic management, and improve environmental sustainability [1], [2]. The emerging IoV services pose stringent requirements on latency, throughput, reliability, and global connectivity, which exceeds the capabilities of current terrestrial networks [3]. To support multifarious IoV services with diversified Quality-of-Service (QoS) requirements, the next-generation (i.e., 5G, 5G beyond, or 6G) network is expected to expand the breadth and depth of communication coverage by integrating nonterrestrial networks to boost network capacity, enhance system robustness, and provide ubiquitous 3-D wireless coverage. Satellite communications, which are globally available and invulnerable to natural disasters, are deemed as a promising solution to provide reliable and ubiquitous high-bandwidth network connectivity in the next-generation networks. With an increasing number of initiatives to construct satellite constellations (e.g., Iridium, Starlink, and Telesat), thousands of low Earth orbit (LEO) satellites will be launched by 2022 to provide communication services for ground users. Furthermore, the dynamic service demands and emergency situations (e.g., service congestion or disaster relief) can be addressed by leveraging the agility of unmanned aerial vehicles (UAVs)-based aerial networks. Therefore, it is essential to develop space–air–ground-integrated networks (SAGINs), which leverage the complementary benefits of all three network segments to ensure seamless, robust, and reliable IoV service provisioning [4].

Despite the benefits brought by SAGIN, it is also challenging for network operators to manage and coordinate the heterogeneous and dynamic physical resources in SAGIN. To accommodate the tremendous traffic volume and highly diversified QoS requirements of IoV services, an agile, flexible, scalable, and cost-effective resource management framework is expected. Network function virtualization (NFV) and software-defined networking (SDN) are recognized as two enabling technologies to address this challenge [5]–[10]. Compared

Manuscript received October 11, 2020; revised December 27, 2020; accepted January 27, 2021. Date of publication February 9, 2021; date of current version April 7, 2022. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada; in part by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS); in part by the National Natural Science Foundation of China under Grant 61801011; and in part by the Beijing Municipal Natural Science Foundation under Grant L192028. This article was presented in part at *IEEE Globecom2019 MWN Symposium,*Waikoloa, HI, USA, December 2019. *(Corresponding author: Weisen Shi.)*

Junling Li is with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Chinese University of Hong Kong, Shenzhen 518100, China, and also with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 5G5, Canada (e-mail: lijunling@cuhk.edu.cn).

Weisen Shi, Huaqing Wu, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 5G5, Canada (e-mail: w46shi@uwaterloo.ca; h272wu@uwaterloo.ca; sshen@uwaterloo.ca).

Shan Zhang is with the School of Computer Science and Engineering, Beihang University, Beijing 100191, China (e-mail: zhangshan18@buaa.edu.cn).

Digital Object Identifier 10.1109/JIOT.2021.3058250

with existing SAGIN architectures, the SDN/NFV-enabled SAGIN architecture has the following major advantages.

1) *Centralized Network Management:* SDN separates the data plane from the control plane and realizes centralized network control. In this way, SDN is able to facilitate service delivery and offer more agility in provisioning both virtual and physical network resources at a central location [9].
2) *Increased Deployment Flexibility of Network Functions:* NFV decouples the network functions from the hardware equipment by virtualizing heterogeneous resources to support diversified network functions [11]. This allows dynamic migration of virtual network functions (VNFs) from one NFV node to another. NFV also simplifies the function deployment process since the function update can be completed by only updating the software without reimplementing devices [12]; For example, privacy-preserving IoV services can be supported by deploying privacy-related VNFs [13].
3) *Reduced Capital and Operational Costs and Improved Resource Utilization:* With the help of SDN, NFV allows a common physical network to support multiple customized services simultaneously. Improved hardware upgrading and maintaining efficiency also leads to reduced capital and operational costs [14].
4) *Better Network Function Provisioning and Isolation:* By taking advantage of NFV, traditional function-specific middleboxes are replaced by high volume commodity servers. In this way, one can deploy a new service by installing VNFs on commodity servers instead of adding new hardware devices. This allows self-configuration and isolation of each function.

With SDN and NFV, an IoV service can be represented by a service function chain (SFC), which is formed with a set of chained VNFs. These VNFs are installed and executed on the heterogeneous network nodes (i.e., NFV nodes), e.g., satellites, UAVs, and vehicles in the SAGIN. For instance, a vehicle monitor/surveillance service can be completed by executing the following network functions in sequence: video recording, video compression, video decoding, subtracting background, object detection, and object classification. These functions can be implemented in different network nodes by considering their capacities and characteristics. Taking video compression for example, it can be deployed flexibly on satellites, UAVs, or vehicles in order to achieve improved resource utilization.

To better support customized IoV services in SDN/NFV-enabled SAGIN, heterogeneous resources have to be allocated to SFCs by the NFV management and orchestration (NFV-MANO), while the SDN controller is placed in a logically centralized location to assign physical links between different VNFs flexibly. This task usually consists of two subtasks: 1) VNF mapping and 2) VNF scheduling. Typically, VNF mapping maps each VNF in an SFC onto a physical node in SAGIN. The locations of NFV nodes to execute the VNFs and routing paths from the source to the destination are simultaneously determined [14]–[17]. After VNF mapping, VNF scheduling finds the most suitable time slots for each service to execute the VNFs mapped onto specific VNF nodes, which improves the overall network performance [18]–[23]. However, the existing works either focus on static scenarios or consider VNF remapping only regardless of finding the VNF scheduling strategy. In practical IoV scenarios, the IoV services arrive to the system dynamically with diversified QoS requirements, and the conditions of the underlying physical infrastructure vary over time. A static service provisioning strategy cannot meet the dynamic properties of IoV services and may lead to inefficient resource utilization. Therefore, dynamic VNF mapping and scheduling algorithms that allow remapping and/or rescheduling of existing VNFs are of great importance to improve the overall performance of IoV service provisioning in SDN/NFV-enabled SAGIN [1].

In this article, we develop efficient online VNF mapping and scheduling algorithms to address the aforementioned challenges. The objective is to maximize the service provider's accumulative profit by readjusting the current mapping and scheduling strategy of VNFs upon new IoV services' arrivals. We first formulate the joint problem of dynamic VNF mapping and scheduling mathematically as a mixed-integer linear programming (MILP) considering VNF mapping constraints, VNF scheduling constraints, and IoV service end-to-end (E2E) delay constraints. Since the problem is known as NP-hard, we then propose two Tabu search (TS)-based algorithms, namely, TS-based pure VNF rescheduling (TS-PSCH) and TS-based VNF remapping and rescheduling (TS-MAPSCH), to obtain near-optimal solutions to the joint problem efficiently. During the VNF remapping process, both the additional delays and migration costs incurred by VNF live migration and reinstantiation are considered. Our main contributions can be summarized as follows.

1) We present an SDN/NFV-based SAGIN architecture to support E2E IoV service deliveries in a dynamic environment, where VNFs can be remapped and/or rescheduled according to the variations of IoV services.
2) Based on this network architecture, we model the migration cost and additional delay incurred by VNF live migration and reinstantiation in the VNF remapping process, considering the heterogeneity of space, air, and ground nodes in the SAGIN. Then, we formulate the joint problem of dynamic VNF mapping and scheduling mathematically incorporating the cost and delay models.
3) To achieve fast and dynamic service deployment in practical online scenarios, we present two TS-based algorithms with low complexity to solve the formulated MILP problem efficiently. The proposed algorithms increase the possibility of new service admissions and thus improve the service provider's profit.
4) Extensive simulation results are provided to demonstrate that the proposed solution is very close to the optimum and that the proposed dynamic algorithms outperform existing works with respect to multiple performance metrics, including total profit, service acceptance ratio, and QoS satisfaction level.

The remainder of this article is organized as follows. In Section II, we provide a review of related works. In Section III, we present the system model in detail and describe the dynamic VNF mapping and scheduling process with an illustrative example. In Section IV, we formulate the

joint dynamic VNF mapping and scheduling problem as an MILP. In Section V, we propose two TS-based algorithms to obtain near-optimal solutions to the MILP problem. In Section VI, simulations are carried out to demonstrate the effectiveness of the proposed TS-based algorithms, followed by the conclusions and future directions in Section VII.

II. Related Works

Motivated by the network performance gain via the virtualization and flexible deployment of network functions, VNF mapping has attracted broad interests with the objective of minimizing the mapping cost or maximizing the revenue [14]–[17]. A comprehensive survey can be found in [12]. One dimension along which the related works can be grouped into is whether the dynamic property of services is considered and the readjustment of the current VNF mapping and scheduling strategies is allowed.

A. Static VNF Mapping and Scheduling

VNF mapping (or VNF embedding) remains a hot research topic in the very recent years [14]–[18]. For example, the study in [14] investigates the joint problem of VNF placement and traffic routing for multicast services. Agarwal *et al.* [15] leveraged the queueing theory to model VNFs and devised a heuristic algorithm to perform VNF mapping and CPU allocation jointly, in order to minimize the ratio between the actual and the maximum allowed latency. In [16], the joint task of virtual node/link mapping and admission control has been first formulated as an MILP and then solved by convex approximation methods after a reformulation. All these works consider static VNF mapping where the adjustment of current mapping strategy is not allowed.

Compared with numerous research activities on VNF mapping, the number of studies on VNF scheduling remains limited. As the first study on VNF scheduling, the joint VNF mapping and scheduling problem is formulated as a flexible job-shop problem that can be solved in two separate stages [19]. Qu *et al.* [21] formulated the joint VNF scheduling and traffic routing problem as an MILP, taking into account the link transmission delay to enable delay-aware VNF scheduling. A genetic algorithm-based heuristic algorithm is then presented to obtain local optimal solutions without QoS consideration. Pham *et al.* [23] presented a game theory-based approach to address deadline-aware VNF scheduling. There exist a few research works that study VNF mapping and scheduling jointly. For example, Alameddine *et al.* [22] studied the deadline-aware VNF scheduling problem and presented an MILP formulation for the joint problem of VNF mapping, traffic routing, and VNF scheduling based on a discrete-time model. The heuristic algorithm solves MILP by separating VNF mapping and VNF scheduling. In [18], three greedy algorithms and a TS metaheuristic algorithm are proposed to support online VNF mapping and scheduling. However, all these works focus on static scenarios, in which the consideration of readjusting the current mapping and scheduling strategies is missing.

B. Standalone Dynamic VNF Mapping

Recently, some research efforts have been devoted to the development of dynamic VNF mapping approaches [24]–[29] that allow VNF remapping/migration mechanisms. For instance, Hawilo *et al.* [25] highlighted that the E2E delay of services will be affected once a VNF is migrated (or reinstantiated) from one hosting NFV node to another. In [29], the problem of dynamic SFC deployment and readjustment is investigated to maximize the service provider's profit. Nevertheless, all the above-mentioned studies consider VNF remapping only regardless of finding the VNF scheduling strategy. In IoV scenarios, the vehicles have high mobility and the IoV services manifest high diversity in service requirements, especially when delay-sensitive and delay-tolerant services coexist in the system. Therefore, readjustment of current scheduling strategies is required for IoV service provisioning to adapt to the dynamic vehicular environments, which falls outside of the capability of existing mechanisms.

C. VNF Mapping and Scheduling in SAGIN

Envisioned as a key enabler for the next-generation networks, the integration of space, air, and ground network segments has attracted substantial research attention. In specific, a comprehensive survey on SAGIN network design and resource allocation is presented in [4]. In [30], artificial intelligence techniques are utilized to address the challenges in the complex SAGIN environment. In [31], the SAGIN is leveraged for maritime coverage enhancement. Focusing on highly mobile/vehicular networks, UAV-assisted service provisioning is investigated in [32] and [33]. The power allocation problem in cognitive satellite-vehicular networks is studied in [34]. Recently, SDN and NFV have also been embraced into the SAGIN to facilitate flexible network control, efficient network configuration, and adaptable resource management [6]–[10]. In [6], SDN and NFV are leveraged to enable agile bidirectional mission offloading in the SAGIN by coordinating the heterogeneous resources from space, air, and ground. Akyildiz and Kak [7] introduced the use of SDN and NFV in the Internet of space things by designing a new class of miniaturized satellites called CubeSats. The CubeSats are then used to form a cyber–physical system, which integrates the ground, air, and space segments. As a further study, the integration of CubeSats into multitenant scenarios is investigated in [8]. In [10], an SFC-based reconfigurable service provisioning framework is presented for the SAGIN. The VNF mapping process is formulated as an integer nonlinear programming (INLP) with the objective of maximizing the number of accommodated service requests while minimizing the cost of running the VNFs on the network nodes. The work in [10] mainly investigates VNF mapping in static scenarios while, in this article, we focus on dynamic VNF mapping that allows the readjustment of the VNF mapping strategy of existing SFCs in the SAGIN. Moreover, the work in [10] focuses only on VNF mapping while we investigate VNF mapping and scheduling jointly to achieve the best performance of SFC-based IoV service provisioning. In summary, although research on static

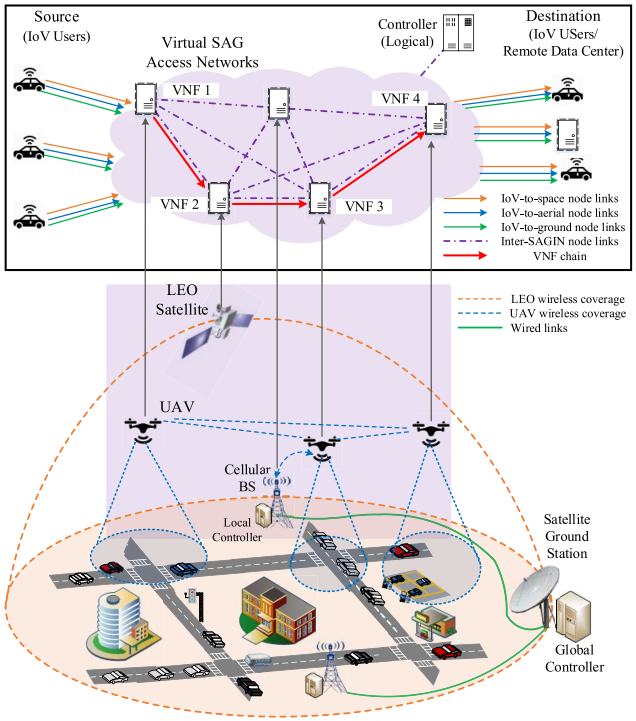

Fig. 1. SDN/NFV-enabled SAGIN scenario for supporting IoV services.

VNF embedding and scheduling have been studied from different aspects, the online VNF rescheduling and remapping in SAGIN is still in its infancy and requires further investigations. Therefore, in this work, we study the readjustment of VNF mapping and scheduling strategies jointly to support multifarious IoV service provisioning in online scenarios for the SAGIN.

III. System Model and Problem Description

As shown in Fig. 1, we consider a general SDN/NFV-enabled SAGIN scenario for supporting IoV services, which consists of space nodes (i.e., LEO satellites) in the space network, aerial nodes (i.e., UAVs) in the air network, and ground nodes (i.e., ground base stations) in the ground network. The LEO satellites have a large coverage and are able to connect to all the network nodes via wireless links. The aerial nodes are mutually connected via directional wireless communication techniques (e.g., mmWave, free-space optical, etc.), and can connect to ground base stations via wireless backhaul links. The ground base stations are interconnected through wired links. An SDN controller is deployed at the satellite ground station (SGS), which can be regarded as a logically centralized location with a global view of the whole network and orchestrates the VNFs to support the IoV services. To perform VNF mapping and scheduling, the SDN controller stores information, such as the physical locations of the NFV nodes, network conditions, services' QoS requirements, resource availability, etc. Note that the SDN controller is logically centralized but can be physically distributed. For large-scale SAGIN, distributed/hierarchical SDN controllers may be deployed to fulfill the tasks of VNF orchestration and resource allocation. The E2E vehicle services are represented by SFCs composed of a predefined order of VNFs. These VNFs are to be mapped onto certain nodes in the SDN/NFV-enabled SAGIN and then scheduled properly in order to support the vehicular services with E2E delay-guarantee. Note that the IoVs are only source or destination nodes of the IoV services in the SDN/NFV-enabled SAGIN and do not carry any VNFs. Each space, aerial, or ground node can be chosen as the initial/last VNF node of an SFC by a source/destination IoV within its coverage. Tables I and II summarize the important parameters and the decision variables used in this article, respectively.

TABLE I
Summary of Important Parameters

Notations	Descriptions
$\mathcal{N}$	set of NFV nodes
$\mathcal{N}_S$	set of space NFV nodes
$\mathcal{N}_A$	set of aerial NFV nodes
$\mathcal{N}_G$	set of ground NFV nodes
c_n	CPU capacity of NFV node $n \in \mathcal{N}$
β_n	buffer capacity of NFV node $n \in \mathcal{N}$
t_c	current service's arrival time (current system time)
$\bar{s}$	newly arrived service at time t_c
$\mathcal{S}_u$	set of unfinished requests in the system at t_c
$\mathcal{F}_s$	set of VNFs in service $s \in \mathcal{S}_u$
D_s	E2E delay requirement of $s \in \mathcal{S}_u$
t_s^a	arrival time of service $s \in \mathcal{S}_u$
t_s^c	completion time of service $s \in \mathcal{S}_u$
δ_{sf}^n	binary variable indicating if VNF $f \in \mathcal{F}_s$ is supported by NFV node n
ρ_{sf}^n	processing time of VNF $f \in \mathcal{F}_s$ on NFV node n
p_{sf}^m	cost for live migrating VNF $f \in \mathcal{F}_s$
p_{sf}^r	cost for re-instantiating VNF $f \in \mathcal{F}_s$
τ_{sfn}^m	additional delay due to live migration to NFV node n for VNF $f \in \mathcal{F}_s$
τ_{sfn}^r	additional delay due to re-instantiation on NFV node n for VNF $f \in \mathcal{F}_s$
$\mathcal{R}_s$	revenue of supporting service s when its E2E delay requirement is just satisfied
ϵ_s	E2E delay violation degree of service s
ϵ_{max}	maximum allowed E2E delay violation degree
$\bar{x}_{sf}^n$	binary constant indicating whether or not VNF $f \in \mathcal{F}_s$ is mapped onto node n before t_c
$\bar{t}_{sf}^b$	beginning time of processing fth VNF in s before t_c
b_{sf}	binary constant indicating whether or not VNF $f \in \mathcal{F}_s$ is being processed upon current service's arrival

A. System Model

Substrate Network: We consider the substrate network as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where the set $\mathcal{N}$ contains the NFV nodes and $\mathcal{L}$ contains all the links among the NFV nodes. The set $\mathcal{N}$ can be further divided into three subsets, i.e., $\mathcal{N}_S$, $\mathcal{N}_A$, and $\mathcal{N}_G$, which contain the space nodes, aerial nodes, and ground base stations in the system, respectively. Notably, we have $\mathcal{N}_S \cup \mathcal{N}_A \cup \mathcal{N}_G = \mathcal{N}$. We assume that the bandwidth and delay properties of each link are adequate to the types of links

TABLE II
SUMMARY OF DECISION VARIABLES

Decision variables	Descriptions
t^b_{sf}	new beginning time of processing VNF $f \in \mathcal{F}_s$ after t_c
x^n_{sf}	binary variable indicating if $f \in \mathcal{F}_s$ is mapped onto NFV node $n \in \mathcal{N}$ after t_c
y^m_{sf}	binary variable indicating if $f \in \mathcal{F}_s$ has live migration after t_c
y^r_{sf}	binary variable indicating if $f \in \mathcal{F}_s$ has re-instantiation after t_c
$z^n_{sf,s'f'}$	binary variable indicating if $f \in \mathcal{F}_s$ starts processing before $f' \in \mathcal{F}'_s$ on node n after t_c

(e.g., LEO-to-ground, UAV-to-ground, UAV-to-UAV, etc.), as well as the mobility pattern of two end nodes. Each NFV node $n \in \mathcal{N}$ is associated with CPU capacity c_n and buffer capacity β_n to host multiple VNFs of different types.

Service Request: We consider that the E2E service requests generated by vehicles arrive to the system dynamically and that the service requests are accommodated in a first-come–first-serve (FCFS) manner Since all the VNF are carried by LEO or UAV nodes whose trajectories are all predefined or predictable, the topology of the virtual SAG network mapped on them is ensured to be highly stable, which can be assumed as quasistationary in a decision-making interval to involve multiple service requests. Each service request s is composed of source, destination, and a sequence of $|\mathcal{F}_s|$ intermediate VNFs, referred to as an SFC. Here, $\mathcal{F}_s$ is the set containing the VNFs in service request s. Note that for each s, its source and destination nodes are randomly selected to model the impacts of end nodes mobility. Denote the E2E delay requirement of s by D_s. Let t^a_s and t^c_s denote the arrival time and completion time (i.e., the time instant when the execution of the last VNF in a service has been finished) of s, respectively. Let the beginning and completion time for processing $f \in \mathcal{F}_s$ be represented by t^b_{sf} and t^c_{sf}, respectively. Let the binary variable $\delta^n_{sf} = 1$ if $f \in \mathcal{F}_s$ can be supported by node n, otherwise, $\delta^n_{sf} = 0$. Each VNF $f \in \mathcal{F}_s$ is associated with a set of NFV nodes $\mathcal{N}_{sf} \subset \mathcal{N}$. Any NFV nodes in the set $\mathcal{N}_{sf}$ has the capability to execute f. When $f \in \mathcal{F}_s$ is being processed or waiting in the queue, a buffer space β_{sf} is required at that NFV node. Let c_{sf} denote the CPU processing resource requirement of $f \in \mathcal{F}_s$. The execution time (i.e., processing delay) for the fth VNF in s on NFV node $n \in \mathcal{N}_{sf}$ is denoted by ρ^n_{sf}. Let the CPU processing capacity of NFV node n for a VNF $f \in \mathcal{F}_s$ be denoted as c^n_{sf}.

Traffic Model: The arrivals of service requests are considered to follow a Poisson distribution. Since multiple VNFs are allowed to be mapped onto a common NFV node, it is assumed that an NFV node can perform packet processing for at most one VNF at a certain time instant.[1] To reduce the switching overhead for VNF scheduling, we assume that a VNF is scheduled for packet processing only if its associated buffer occupancy is above a threshold B. Hence, the number of packets to be processed for one-time VNF scheduling is B, and the packet processing time at VNF $f \in \mathcal{F}_s$ on NFV node n ($n \in \mathcal{N}_{sf}$) for service s is given by

$$\rho^n_{sf} = B/c^n_{sf}. \tag{1}$$

Dynamic VNF Mapping and Scheduling: Suppose that a new service s_c arrives to the system at time t_c and that the set of services being supported by the system at t_c (including the newly arrived one) is $\mathcal{S}_u$. Due to the stringent QoS requirement of the new service, the total revenue obtained from directly admitting the new service without performing any adjustments to the current strategy of VNF mapping and scheduling may not be optimal. The dynamic VNF mapping process is to remap and/or reschedule the unfinished VNFs of existing services in the system so that the service provider's accumulative profit is maximized. The output of dynamic VNF mapping and scheduling includes three parts: 1) for each service $s \in \mathcal{S}_u$, the new VNF mapping strategy after t_c; 2) for each service $s \in \mathcal{S}_u$, the time when the packets start being processed at each of its unfinished VNFs after t_c; and 3) for each NFV node n, the scheduling sequence of all the VNFs mapped onto it after t_c. The online decision-making interval for dynamic adjustment of the VNF mapping and scheduling strategies should be determined carefully based on the network dynamics, services' requirements, operational cost, etc. In this study, the online decision making for dynamic SFC mapping and scheduling is made when the newly arrived service's QoS requirement cannot be strictly satisfied, in order to balance the tradeoff between cost and performance.

Cost and Delay Models for VNF Remapping: To achieve dynamic VNF mapping and scheduling, we consider that the existing VNFs in the system are allowed to be remapped onto new NFV nodes during a VNF remapping and rescheduling process. Depending on if a VNF is being processed at the current service time, two types of VNF migration mechanisms are considered in this study, namely, VNF live migration and VNF reinstantiation [25]. The former refers to the process of migrating the status as well as the data traffic of a VNF being processed from one hosting NFV node to another. The latter is the process of removing a VNF waiting to be processed in the buffer on a hosting NFV node and reinstantiate a new VNF of the same type onto another node. For both cases, additional delay and reconfiguration/migration costs will be incurred due to data transfer between two NFV nodes and additional signaling between the SDN controller and the affected NFV nodes. Considering the heterogeneity of NFV nodes in the SAGIN, the additional delay caused by VNF migration varies for different types of nodes. Therefore, we denote the data transfer time for VNF live migration to node n and the setup time for VNF reinstantiation on node n of the fth VNF in service s by τ^m_{sfn} and τ^r_{sfn}, respectively. In this article, we assume that the live migration delay on a space node is greater than that on an aerial node, which is higher than that on a ground node, i.e., $\tau^m_{sfn}(n \in \mathcal{N}_S) > \tau^m_{sfn}(n \in \mathcal{N}_A) > \tau^m_{sfn}(n \in \mathcal{N}_G)$. Similar assumptions are made for τ^r_{sfn}. Denote the live migration cost and reinstantiation cost for the fth VNF in service s by p^m_{sf} and p^r_{sf}, respectively.

[1] In practical systems, the computing resources on each NFV node are usually indivisible. Each NFV node can only support at most one VNF at a time and all the computing resources are allocated to VNF for packet processing [12], [21], [23], [36], [37].

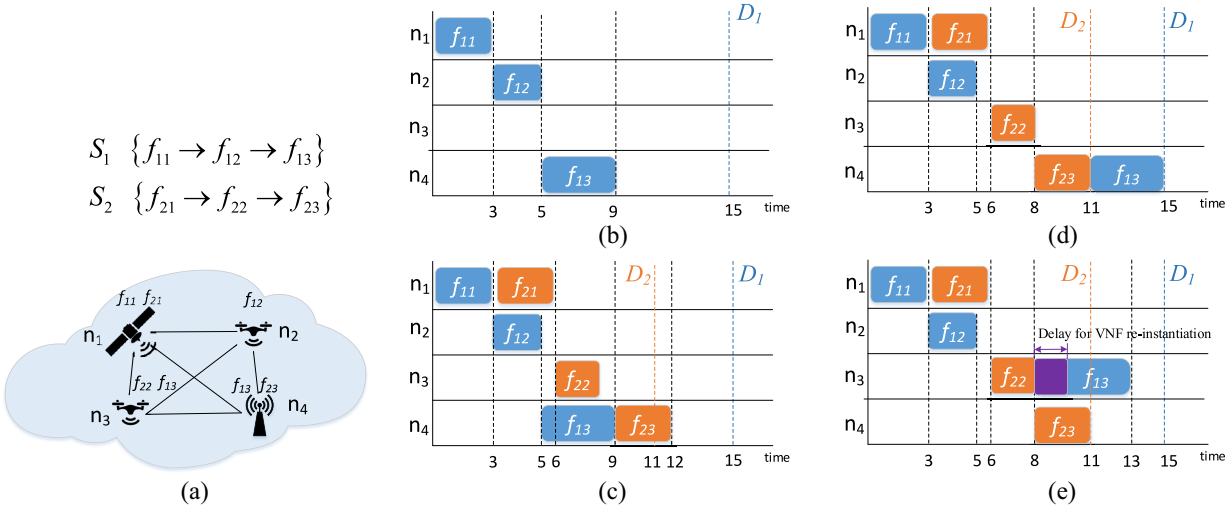

Fig. 2. Illustration of dynamic VNF mapping and scheduling. (a) Sevice requests and substrate network. (b) Initial VNF mapping and scheduling strategy for S_1. (c) Initial VNF mapping and scheduling strategy for S_1 and S_2. (d) Dynamic VNF scheduling strategy for S_1 and S_2. (e) Dynamic VNF mapping and scheduling strategy for S_1 and S_2.

B. Problem Description

Consider a substrate network with limited physical resources, including processing resources, buffer resources, etc. Composed by a sequence of VNFs, the service requests arrive to the SDN/NFV-enabled SAGIN dynamically, with a set of physical resource and E2E delay requirements. Via dynamic VNF instantiation and scheduling, our purpose is to seek for the optimal VNF mapping and scheduling strategy for the VNFs in the services to maximize the acceptance ratio and the service provider's accumulative profit in the long run.

Here, we use a simple example to illustrate the dynamic VNF mapping and scheduling process, as shown in Fig. 2. Consider two service requests, i.e., $s_1 = \{f_{11} \rightarrow f_{12} \rightarrow f_{13}, D_1\}$ and $s_2 = \{f_{21} \rightarrow f_{22} \rightarrow f_{23}, D_2\}$, and a four-node substrate network, as shown in Fig. 2(a). The network nodes have different capabilities to support VNFs. Specifically, n_1 can support f_{11} and f_{21}, n_2 can support f_{12}, n_3 has the capability to support f_{13} and f_{22}, and n_4 can support f_{23} and f_{13}. At time instant $t = 0$, service request s_1 arrives to the system with E2E delay requirement $D_1 = 15$ (time units). On possible initial VNF mapping and scheduling strategy for s_1 is given by Fig. 2(b). At time instant $t = 3$, service request s_2 arrives to the system with E2E delay requirement $D_2 = 8$ (time units), with its initial VNF mapping and scheduling strategy being presented in Fig. 2(c). We can see from Fig. 2(c) that the initial VNF mapping strategy for the two services results in the service completion time of s_2 being $t = 12$. Therefore, the delay requirement of s_2 is violated with this initial mapping and scheduling strategy. Consequently, service s_2 will be rejected if a static VNF mapping algorithm (such as those in [18] and [22]) is used. To enable dynamic VNF mapping and scheduling, in this study, whenever a service is rejected due to delay violation with the initial scheduling strategy, a VNF remapping and rescheduling mechanism is triggered to remap and/or reschedule the unfinished VNFs in the substrate network. Fig. 2(d) shows an example of rescheduling the unfinished VNFs to avoid delay violation of the newly arrived service s_2. The processing sequence of VNFs f_{13} and f_{23}, which are both mapped onto node n_4, is swapped so that the E2E delay requirements of s_1 and s_2 can be satisfied simultaneously. Fig. 2(e) shows an example of remapping and rescheduling the unfinished VNFs. In particular, the VNF f_{13} from s_1 is reinstantiated on n_3. This process incurs an additional delay of 2 time units (labeled by purple). As a result, f_{23} can start processing at $t = 8$, while f_{13} can start processing at $t = 10$. Notably, VNF remapping also reduces the service completion time of s_1, compared with the solution with VNF rescheduling only in Fig. 2(d).

We remark that dynamic VNF mapping and scheduling are beneficial in the following regards: 1) increasing acceptance ratio of the substrate network to accommodate more services, and thus leading to a higher profit; 2) potentially reducing the completion time of the service requests, which enhances the QoS; and 3) better balancing the traffic load among the NFV nodes in the substrate network.

IV. Problem Formulation

In this section, we present the formulation for the dynamic VNF mapping and scheduling problem, where both VNF remapping and VNF rescheduling are allowed and the additional delays and migration costs incurred by VNF live migration and reinstantiation are considered.

Denote the set of service requests in the system at time instant t_c when VNF remapping and rescheduling is triggered by $\mathcal{S}_u$, which contains all the unfinished services that arrive to the system prior to or exactly at t_c, i.e., $\mathcal{S}_u = \{s | t_s^a \leq t_c \text{ and } t_s^c > t_c\}$. In this study, we assume that the E2E delay requirement of a network service can be violated by a certain degree and consider the latest allowed time to finish the processing of the last VNF as the deadline for a given service request. Let $\epsilon_{\max}$ represent the maximum E2E delay violation degree that can be tolerated. Let the delay requirement of s be denoted by D_s. The deadline for service s with arrival time t_s^a is found as $t_s^a + D_s(1 + \epsilon_{\max})$. Let $\mathcal{R}_s$ represent the

service provider's revenue for supporting service $s \in \mathcal{S}_u$ if the delay requirement of s is just satisfied (i.e., $t_s^c - t_s^a = D_s$). The actual revenue for supporting s depends on how well the QoS requirement of s is satisfied and may vary over time due to the dynamic operations.

We define a binary constant $\bar{x}_{sf}^n$ and a binary variable x_{sf}^n to represent the VNF mapping strategy before and after current service time t_c, respectively. Let $\bar{x}_{sf}^n = 1$ if VNF $f \in \mathcal{F}_s$ is mapped onto node n before t_c and let $\bar{x}_{sf}^n = 0$ otherwise. Let $x_{sf}^n = 1$ if VNF $f \in \mathcal{F}_s$ is mapped onto node n after t_c and let $x_{sf}^n = 0$ otherwise. Similarly, we use a parameter $\bar{t}_{sf}^b$ and a continuous variable t_{sf}^b to denote the beginning time of processing the fth VNF in service s before and after current service time t_c, respectively. The overall objective of dynamic VNF remapping and rescheduling is to maximize the service provider's total profit $\mathcal{P}$ at t_c

$$\max_{x_{sf}^n, z_{sf,s'f'}^n, t_{sf}^b} \mathcal{P} = \mathcal{R} - \mathcal{C} \tag{2}$$

where $\mathcal{R}$ represents the sum of revenues of all the unfinished services in the system and $\mathcal{C}$ denotes the sum of migration costs incurred by VNF live migration and reinstantiation in the VNF remapping process. The total revenue $\mathcal{R}$ is found as follows:

$$\mathcal{R} = \sum_{s \in \mathcal{S}_u} \mathcal{R}_s \frac{\epsilon_{\max} - \epsilon_s}{\epsilon_{\max}} \tag{3}$$

where $\epsilon_{\max}$ denotes the maximum E2E delay violation degree that can be tolerated by a service request, and ϵ_s represents the E2E delay violation degree of service s after t_c, given by

$$\epsilon_s = \frac{t_s^c - t_s^a - D_s}{D_s}. \tag{4}$$

The total migration cost $\mathcal{C}$ consists of two parts: 1) the migration cost incurred by VNF live migration and 2) the migration cost due to VNF reinstantiation, given as follows:

$$\mathcal{C} = \sum_{s \in \mathcal{S}_u \backslash \bar{s}} \sum_{f \in \mathcal{F}_s} \left(p_{sf}^m y_{sf}^m + p_{sf}^r y_{sf}^r \right) \tag{5}$$

where y_{sf}^r is a binary variable obtained by integrating $\bar{x}_{sf}^n$ and x_{sf}^n and used to indicate if reinstantiation is needed for VNF $f \in \mathcal{F}_s$ in the new VNF mapping strategy, i.e.,

$$y_{sf}^r = \sum_{n \in \mathcal{N}} \left(1 - \bar{x}_{sf}^n\right) x_{sf}^n \tag{6}$$

where $s \in \mathcal{S}_u \backslash \bar{s}$. Similarly, y_{sf}^m is a binary variable obtained by integrating $\bar{x}_{sf}^n$, x_{sf}^n, and b_{sf}, i.e.,

$$y_{sf}^m = \sum_{n \in \mathcal{N}} \left(1 - \bar{x}_{sf}^n\right) x_{sf}^n b_{sf} \quad \forall\ s \in \mathcal{S}_u \backslash \bar{s}. \tag{7}$$

The binary variable y_{sf}^m is used to indicate if live migration is required for the VNF $f \in \mathcal{F}_s$ in the new VNF mapping strategy. Note that we have $y_{sf}^r = 0$ and $y_{sf}^m = 0$ for the newly arrived service $\bar{s}$ since its mapping and scheduling will incur no VNF live migration and reinstantiation.

The constraints that should be satisfied in the dynamic mapping and scheduling process can be divided into three parts, detailed as follows.

1) VNF Mapping Constraints: First, the following constraint has to be satisfied to ensure that each VNF can only be mapped onto one NFV node

$$\sum_{n \in \mathcal{N}} x_{sf}^n = 1 \quad \forall f \in \mathcal{F}_s\ \forall\ s \in \mathcal{S}_u. \tag{8}$$

Then, for any VNF in a service request, the NFV node onto which it is mapped has to be able to execute it

$$\sum_{n \in \mathcal{N}} \delta_{sf}^n x_{sf}^n = 1 \quad \forall f \in \mathcal{F}_s\ \forall\ s \in \mathcal{S}_u. \tag{9}$$

Finally, for each NFV node in the system, the physical resources occupied by the VNFs mapped onto that node cannot exceed its capacity, i.e.,

$$\sum_{s \in \mathcal{S}_u} \sum_{f \in \mathcal{F}_s} x_{sf}^n c_{sf} \le c_n \quad \forall n \in \mathcal{N} \tag{10}$$

$$\sum_{s \in \mathcal{S}_u} \sum_{f \in \mathcal{F}_s} x_{sf}^n \beta_{sf} \le \beta_n \quad \forall n \in \mathcal{N}. \tag{11}$$

2) VNF Scheduling Constraints: Apart from the above constraints (8)–(11), the following constraints need to be satisfied to have a feasible VNF schedule. First, the specified processing sequence of the VNFs in any service s should be guaranteed

$$t_{s,f+1}^b - t_{sf}^b \ge \rho_{sf} + \tau_{sf}^m y_{sf}^m + \tau_{sf}^r y_{sf}^r \quad \forall f \in \mathcal{F}_s\ \forall\ s \in \mathcal{S}_u \tag{12}$$

where ρ_{sf} denotes the processing time for VNF $f \in \mathcal{F}_s$ in the new scheduling strategy, found by

$$\rho_{sf} = \sum_{n \in \mathcal{N}} x_{sf}^n \rho_{sf}^n \quad \forall f \in \mathcal{F}_s\ \forall\ s \in \mathcal{S}_u. \tag{13}$$

In (12), τ_{sf}^m and τ_{sf}^r denote the additional delay incurred by VNF live migration and reinstantiation for VNF $f \in \mathcal{F}_s$ ($\forall\ s \in \mathcal{S}_u \backslash \bar{s}$), respectively, obtained from

$$\tau_{sf}^m = \sum_{n \in \mathcal{N}} \tau_{sfn}^m x_{sf}^n \tag{14}$$

$$\tau_{sf}^r = \sum_{n \in \mathcal{N}} \tau_{sfn}^r x_{sf}^n. \tag{15}$$

Then, for any two VNFs $f \in \mathcal{F}_s$ and $f' \in \mathcal{F}_{s'}$ mapped onto the common NFV node, the following constraints need to be satisfied to ensure that no preemption takes place at any time:

$$t_{sf}^b + z_{sf,s'f'}^n \left(\rho_{sf} + \tau_{sf}^m y_{sf}^m + \tau_{sf}^r y_{sf}^r \right) \le t_{s'f'}^b + z_{s'f',sf}^n M \tag{16}$$

$$t_{s'f'}^b + z_{s'f',sf}^n \left(\rho_{s'f'} + \tau_{s'f'}^m y_{s'f'}^m + \tau_{s'f'}^r y_{s'f'}^r \right) \le t_{sf}^b + z_{sf,s'f'}^n M \tag{17}$$

$$z_{sf,s'f'}^n + z_{s'f',sf}^n = 1 \tag{18}$$

where $s' \in \mathcal{S}_u$, $f' \in \mathcal{F}_{s'}$, $s \neq s'$ or $f \neq f'$, and M is a big positive number [14]. $z_{sf,s'f'}^n$ is an auxiliary binary variable indicating the processing sequence of the two VNFs mapped onto the same NFV node. $z_{sf,s'f'}^n = 1$ if $f \in \mathcal{F}_s$ starts processing before $f' \in \mathcal{F}_s'$ on NFV node n and $z_{sf,s'f'}^n = 0$ otherwise.

3) E2E Delay Constraints: The following constraint guarantees that the deadline of each unfinished service is not violated:

$$t_s^c - t_s^a \leq D_s(1+\epsilon_{\max}) \quad \forall\ s \in \mathcal{S}_u \tag{19}$$

where t_s^c represents the completion time of service s in the new VNF scheduling strategy, given by

$$t_s^c = t^b_{s|\mathcal{F}_s|} + \rho_{s|\mathcal{F}_s|} + \tau^m_{s|\mathcal{F}_s|} y^m_{s|\mathcal{F}_s|} + \tau^r_{s|\mathcal{F}_s|} y^r_{s|\mathcal{F}_s|} \quad \forall\ s \in \mathcal{S}_u. \tag{20}$$

Now, we formulate the problem of dynamic VNF mapping and scheduling as follows:

$$\begin{aligned} \max_{x^n_{sf}, z^n_{sf,s'f'}, t^b_{sf}} \quad & \mathcal{P} = \mathcal{R} - \mathcal{C} \\ \text{s.t.} \quad & (8)-(12),\ (16)-(19) \\ & t^b_{sf} \geq t_c \\ & x^n_{sf} \in \{0,1\} \\ & z^n_{sf,s'f'} \in \{0,1\}. \end{aligned} \tag{21}$$

Note that this is an MILP problem with continuous variables t^b_{sf} and integer variables x^n_{sf} and $z^n_{sf,s'f'}$. It is known as an *NP*-hard combinatorial optimization problem whose optimal solution cannot be found within a polynomial time [22]. For service provisioning in the highly dynamic IoV scenarios, it is inefficient to solve MILP frequently since the high computational complexity will result in excessive queuing delay for the service requests. To keep pace with the fast-changing vehicular environment and achieve fast service provisioning, we propose two TS-based heuristic algorithms for agile and dynamic VNF mapping and scheduling in the following section.

V. Tabu Search-Based Heuristic Algorithms

In this section, we propose two TS-based heuristic algorithms, namely, TS-MAPSCH and TS-PSCH, to obtain near-optimal solutions to the MILP problem efficiently. Both the algorithms consist of two stages. In the first stage, we adopt a greedy algorithm to obtain fast initial VNF mapping and scheduling strategy upon the arrival of new services. In the second stage, we verify whether or not the E2E delay requirements of the services are strictly satisfied, and trigger a delay-aware heuristic algorithm to remap and reschedule (or only reschedule) the unfinished VNFs in existing services if needed. In what follows, we describe the two proposed algorithms in detail.

A. Greedy Algorithm for Initial VNF Mapping and Scheduling

In this stage, a greedy earliest processing (GEP) algorithm is adopted to obtain a fast initial VNF mapping and scheduling strategy. In specific, upon the arrival of a new service request $\bar{s}$, its VNFs $\mathcal{F}_{\bar{s}}$ are mapped onto the NFV nodes sequentially. For each VNF $f \in \mathcal{F}_{\bar{s}}$, a number of candidate NFV nodes $\bar{\mathcal{N}}_{\bar{s}f} \subset \mathcal{N}_{\bar{s}f}$ are selected, in which each node has the ability and enough CPU and buffer resources to support f. The service will be rejected if no candidate node can be found in $\mathcal{N}_{\bar{s}f}$. Then, the candidate nodes for each VNF are sorted according to their earliest available start time, i.e., the time instant when the packet processing can start for f if f is mapped onto the candidate node. The NFV node that provides the earliest start time for f will be chosen to support f [18].

Algorithm 1: *Slackness*-Based Algorithm for Finding the Initial Solution π_0 of the TS Procedure

Input: Newly arrived service $\bar{s}$, current system time t_c
Output: Initial solution π_0 to the TS procedure

```
Add s̄ to S_u;
Calculate the Slackness value for s̄: Δ_s̄ = D_s̄ − (t^c_s̄ − t^a_s̄);
for s ∈ S do
    if t^a_s ≤ t_c and t^c_s > t_c then
        Add s to S_u;
        Calculate the Slackness value for s: Δ_s = D_s − (t^c_s − t^a_s);
Sort the existing services in S_u according to Δ_s;
for s ∈ S_u do
    Remap and reschedule the unfinished VNFs in s sequentially using the GEP algorithm;
Set π_0 ← the new VNF mapping and scheduling strategy;
return π_0;
```

The service completion time $t^c_{\bar{s}}$ for the newly arrived service $\bar{s}$ can be obtained immediately after a successful initial VNF mapping and scheduling is done. Next, the heuristic algorithm checks whether the delay requirement of $\bar{s}$ can be strictly satisfied. If $D_{\bar{s}} \geq t^c_{\bar{s}} - t^a_{\bar{s}}$, the newly arrived service will be directly admitted with corresponding CPU processing and buffer resources being allocated. Otherwise, the proposed TS-based algorithms for VNF remapping and rescheduling will be triggered, which will be detailed in subsequent sections.

B. Proposed TS-MAPSCH Algorithm

In this section, we propose a TS-MAPSCH algorithm to seek for a better processing sequence of existing services and the corresponding VNF mapping and scheduling strategy with a TS procedure. A TS-based algorithm typically consists of five components, i.e., initial solution, neighboring solutions, Tabu list [38], aspiration criteria, and stopping condition. In what follows, we describe these concepts in the context of the proposed TS-MAPSCH algorithm. More details of the algorithm are shown in Algorithm 2.

1) Initial Solution: Essentially, any mapping and scheduling strategies that can satisfy all the constraints in problem (21) can be considered as an initial solution of the TS procedure. However, as a local search algorithm, the final solution obtained from the TS procedure highly relies on the quality of the initial solution. Let π_0 represent the initial solution of the TS procedure. To have a promising π_0, in this study, π_0 is obtained as follows. First, we define and calculate the *Slackness* Δ_s, for each service $s \in \mathcal{S}_u$, as $\Delta_s = D_s - (\bar{t}^c_s - t^a_s)$ where $\bar{t}^c_s$ represents the service completion time of s before current service's arrival. The *Slackness* of an existing service

Algorithm 2: Proposed TS-MAPSCH Algorithm

1 Obtain the initial solution π_0 for the TS procedure using Algorithm 1;
2 Calculate the profit for the initial solution $\mathcal{P}(\pi_0)$;
3 **if** π_0 *is not feasible* **then**
4 Set *reMapReSchSuccess* $\leftarrow$ *false*;
5 **return**;
6 Set $\pi^* \leftarrow \pi_0$, $\pi \leftarrow \pi_0$, $\mathcal{P}(\pi^*) \leftarrow \mathcal{P}(\pi_0)$;
7 Initialize *TabuList* as an empty list;
8 **while** *stopConditionNotMet*() **do**
9 *Neighbors* $\leftarrow$ *getNeighbors*(π);
10 Set the profit of the best neighbor $\mathcal{P}_n^* \leftarrow 0$;
11 **for** π' *in Neighbors* **do**
12 **if** π' *is feasible* **then**
13 Perform VNF remapping and rescheduling for π' using GEP;
14 Calculate the total profit $\mathcal{P}(\pi')$ for π';
15 **if** $\mathcal{P}(\pi') > \mathcal{P}_n^*$ *and* π' *is not in TabuList* **then**
16 Set $\pi_n^* \leftarrow \pi'$;
17 Set $\pi \leftarrow \pi_n^*$;
18 **if** $\mathcal{P}_n^* > \mathcal{P}(\pi^*)$ **then**
19 Set $\pi^* \leftarrow \pi_n^*$;
20 *updateTabuList*(π_n^*);
21 Set *reMapReSchSuccess* $\leftarrow$ *true*;
22 **return** π^*;

intuitively represents the sensitive degree of the service to the E2E delay. A small *Slackness* implies that the service has limited spaces for VNF remapping and rescheduling and also indicates that s should be given a higher priority in the VNF remapping and rescheduling process. Then, all the existing services in $\mathcal{S}_u$ are sorted in ascending order according to Δ_s. After that, for each service in $\mathcal{S}_u$, the unfinished VNFs are remapped and rescheduled sequentially using the GEP algorithm. The resultant VNF mapping and scheduling strategy is considered as the initial solution of the TS-MAPSCH algorithm. The *Slackness*-based algorithm for finding the initial solution π_0 is shown in Algorithm 1.

2) Neighboring Solutions: In the TS procedure, each neighbor of a current solution, π', can be obtained as follows: 1) randomly choose two services from $\mathcal{S}_u$ and swap the processing sequence of them; 2) remap and reschedule the unfinished VNFs in the existing services sequentially using the GEP algorithm with the new service processing sequence (line 13); and 3) set the new mapping and scheduling strategy π' as one of the neighbors of the current solution. After all the neighbors are found, we evaluate the total profit for each neighbor $\mathcal{P}(\pi')$ and determine if the best solution seen to date π^* and its corresponding profit $\mathcal{P}(\pi^*)$ needs to be updated (lines 14–16).

3) Tabu List: If the processing sequence of two existing services has been swapped, we declare it as Tabu to swap the processing sequence of them again in the next L iterations, where L is the length of the Tabu list. It is also Tabu to allow the current solution to move to a solution in which the QoS violation degree of an existing service exceeds the maximum tolerance $\epsilon_{\max}$. In other words, all the neighboring solutions found in each iteration should result in a feasible VNF schedule without violating the E2E delay requirement of any existing services.

4) Aspiration Criterion: We do not consider it as a Tabu move if the new solution results in a higher profit than the highest profit known to date. In other words, if $\mathcal{P}(\pi') > \mathcal{P}(\pi^*)$, we consider π' as a feasible solution even if the movement from the current solution π to π' is contained in the Tabu list.

5) Stopping Criteria: The TS procedure stops if the iteration counter reaches its maximum value or the best solution found by the algorithm remains unchanged in a certain number of successive iterations.

Clearly, the TS-MAPSCH algorithm allows the SAGIN to admit the newly arrived service by readjusting the QoS violation degrees of the existing services. Therefore, the total profit of supporting the existing services may decrease after the readjustment. This is due to that the migration cost incurred by VNF live migrations and reinstantiations may be significant, especially in high-migration-cost scenarios. To maximize the service provider's profit in the long run, we have to determine if the new mapping and scheduling strategy π^* found by the TS procedure deserves to be executed. This can be done by comparing the total profit obtained from executing π^* and that from admitting the new service directly without triggering the VNF remapping and rescheduling mechanism.

Remark 1: The TS-MAPSCH algorithm is computationally efficient as the dynamic VNF mapping and scheduling can be made in polynomial time.

Each initial VNF mapping and scheduling involves one time sorting of $|\mathcal{N}(f_{sf})|$ NFV nodes' available start times, whose typical computational complexity is $O(|\mathcal{N}(f_{sf})| \log |\mathcal{N}(f_{sf})|)$. For the TS-MAPSCH algorithm, to obtain its initial solution π_0, we first sort $|\mathcal{S}_u|$ unfinished services according to their *Slackness* values, which has a computational complexity of $O(|\mathcal{S}_u| \log |\mathcal{S}_u|)$. Let the number of neighbors in each iteration be denoted by M and the maximum allowed number of iterations be denoted by K. For each neighboring solution in the TS procedure, the VNF remapping and rescheduling strategy is obtained by using the GEP algorithm. Therefore, the total computational complexity to find the best solution is $O(KM|\mathcal{N}(f_{sf})| \log |\mathcal{N}(f_{sf})|)$. Hence, the proposed TS-MAPSCH algorithm is an efficient polynomial-time algorithm.

C. Proposed Tabu Search-Based Pure VNF Rescheduling Algorithm

The proposed TS-PSCH algorithm performs in a similar manner as the TS-MAPSCH algorithm. The difference between the two algorithms lies in whether VNF remapping is allowed or not during the TS procedure. The details of the TS-PSCH algorithm are shown in Algorithm 3. Specifically, given a processing sequence of the existing services in the TS procedure, the TS-PSCH algorithm reschedules the VNFs in $\mathcal{S}_u$ sequentially (line 13). During this process, the current

Algorithm 3: Proposed TS-PSCH Algorithm

Obtain the initial solution π_0 for the TS procedure using Algorithm 1;
Calculate the profit for the initial solution $\mathcal{P}(\pi_0)$;
if π_0 *is not feasible* **then**
    Set *reSchSuccess* $\leftarrow$ *false*;
    return;
Set $\pi^* \leftarrow \pi_0$, $\pi \leftarrow \pi_0$, $\mathcal{P}(\pi^*) \leftarrow \mathcal{P}(\pi_0)$;
Initialize *TabuList* as an empty list;
while *stopConditionNotMet*() **do**
    Neighbors $\leftarrow$ *getNeighbors*(π);
    Set the profit of the best neighbor $\mathcal{P}_n^* \leftarrow 0$;
    for π' *in Neighbors* **do**
        if π' *is feasible* **then**
            Perform pure VNF rescheduling for π';
            Calculate the total profit $\mathcal{P}(\pi')$ for π';
            if $\mathcal{P}(\pi') > \mathcal{P}_n^*$ *and TabuList.contains*(π') *is false* **then**
                Set $\pi_n^* \leftarrow \pi'$;
    Set $\pi \leftarrow \pi_n^*$;
    if $\mathcal{P}_n^* > \mathcal{P}(\pi^*)$ **then**
        Set $\pi^* \leftarrow \pi_n^*$;
    updateTabuList(π_n^*);
Set *reSchSuccess* $\leftarrow$ *true*;
return π^*;

VNF mapping strategy remains the same but the beginning time for processing existing VNFs may be changed due to the swap of the processing sequence of existing services. This mechanism will neither incur additional delay to any of the existing services nor incur additional migration costs to the service operator as no VNF live migration and reinstantiation is required. However, since the VNFs are not allowed to migrate from one NFV node to another, the possibility of finding the optimal VNF mapping and scheduling strategy during the TS procedure is greatly reduced. Therefore, to achieve cost-effective dynamic VNF remapping and rescheduling, one has to carefully balance the tradeoff between migration cost and network performance.

Remark 2: The proposed TS-MAPSCH algorithm allows both VNF live migrations and VNF reinstantiations in remapping the VNFs of existing services. Therefore, compared with TS-PSCH, TS-MAPSCH offers a higher chance for the TS algorithm to seek for the best VNF mapping and scheduling strategy and achieves more flexible VNF placement in dynamic network environments. However, the more VNF migrations/reinstantiations allowed, the larger migration cost and additional delay will be incurred in executing the new strategy. Therefore, there is a tradeoff between increasing the total profit and decreasing the migration cost. Simulation results presented in the following section imply that in scenarios where the migration cost is relatively low, TS-MAPSCH outperforms TS-PSCH, while when the migration cost for VNF migrations/reinstantiations becomes higher, TS-PSCH may have a better performance.

TABLE III
PARAMETER RANGES IN SIMULATION SCENARIOS

Parameter	Minimum	Maximum
CPU capacity for each NFV node	50	100
Buffer capacity for each NFV node	50	100
Number of VNFs supported per node	1	5
VNF processing time (time units)	5	10
CPU requirement of each VNF	10	20
Buffer requirement of each VNF	10	20
Number of VNFs in each request	3	5
Maximum allowed delay violation ϵ_{max}	0.1	1.0
E2E delay requirement D_s	20	125
Revenue of service requests $\mathcal{R}_s$	50	100
Service arrival rate	0.05	0.2

VI. PERFORMANCE EVALUATION

In this section, the performance of the two proposed TS-based algorithms is evaluated and compared with that of two existing algorithms [1], [18], and the MILP model presented in Section IV. We refer to the work [18] as the static benchmark algorithm whose idea can be described as follows: upon the arrival of a new service request, the static benchmark algorithm maps and schedules VNFs in SFC using the greedy algorithm described in Section V-A. If the E2E delay violation can be tolerated, the service is admitted and allocated with corresponding physical resources. Otherwise, the new service is rejected. The work in [1] is referred to as the slackness greedy algorithm in which the unfinished services are first sorted according to their slackness values, and then remapped and rescheduled greedily. For the MILP model, we employ the Gurobi optimization solver to obtain the optimal solution. For the proposed TS-based algorithms, we set the length of the Tabu list to be 10 (i.e., $L = 10$), the number of neighbors to be 30 (i.e., $M = 30$), and the maximum number of TS iterations to be 50 (i.e., $K = 50$).

In our simulation, we consider an SAGIN scenario with ten heterogeneous network nodes, which consists of one space node, three aerial nodes, and six ground base stations. The additional delays due to VNF live migration to the space nodes, the aerial nodes, and the ground base stations are set as $\tau_{sfn}^m = 4$ ($n \in \mathcal{N}_S$), $\tau_{sfn}^m = 3$ ($n \in \mathcal{N}_A$), and $\tau_{sfn}^m = 2$ ($n \in \mathcal{N}_G$) (all in time units), respectively. The additional delays due to VNF reinstantiation on the space nodes, aerial nodes, and ground base stations are set as $\tau_{sfn}^r = 3$ ($n \in \mathcal{N}_S$), $\tau_{sfn}^r = 2$ ($n \in \mathcal{N}_A$), and $\tau_{sfn}^r = 1$ ($n \in \mathcal{N}_G$) (all in time units), respectively. We consider that the service requests arrive to the system according to a Poisson distribution. The service arrival rate is in the range of [0.05, 0.2] (service per time unit). Other parameters are summarized in Table III.

A. Low-Migration-Cost Online Scenario

1) Acceptance Ratio and Total Profit: We first consider a low-migration-cost scenario and demonstrate the effectiveness of the proposed dynamic VNF mapping and scheduling algorithms in terms of increasing the service acceptance ratio

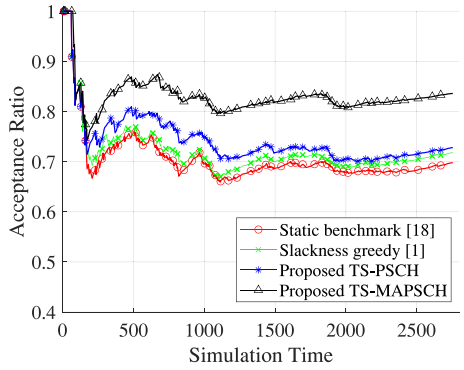

Fig. 3. Comparison of acceptance ratio between the two existing algorithms and the proposed TS-based algorithms in the low-migration-cost scenario.

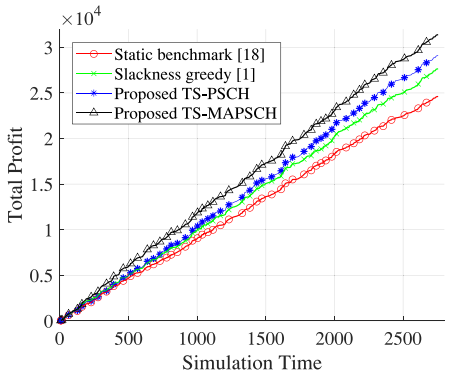

Fig. 4. Comparison of total profit between the two existing algorithms and the proposed TS-based algorithms in the low-migration-cost scenario.

and total profit. To this end, we set the cost for VNF live migration and reinstantiation as $p^m_{sf} = 0.4$ and $p^r_{sf} = 0.2$, respectively. The simulation is conducted with 500 services and the service arrival rate is set as 0.2 service per time unit. The maximum E2E delay violation degree is set as 50% for all the service requests (i.e., $\epsilon_{\max} = 0.5$). The comparison of the four algorithms in terms of service acceptance ratio is shown in Fig. 3. We can see from the figure that the TS-PSCH performs slightly better than the two existing algorithms, while the TS-MAPSCH algorithm maintains a much higher acceptance ratio over the other three algorithms. On average, the proposed TS-MAPSCH algorithm admits about 10% and 15% more services than TS-PSCH and the static mapping and scheduling solution, respectively. The reason is that the TS-MAPSCH triggers both VNF remapping and VNF rescheduling mechanisms and thus offers a larger degree of freedom for the algorithm to obtain better service provisioning strategy than TS-PSCH (in which only VNF rescheduling is triggered) and the static benchmark (in which neither the current VNF mapping nor the current VNF scheduling strategy can be readjusted). Similar conclusion can be drawn from the total profit comparison in Fig. 4, where the four algorithms have similar performance at the beginning of the simulation. But in the end, the TS-MAPSCH and TS-PSCH algorithms achieve approximately 30% and 20% (15% and 5%) higher total profit over the static benchmark algorithm (the slackness greedy algorithm), respectively. We can also conclude that when the cost for VNF migrations is relatively low, the

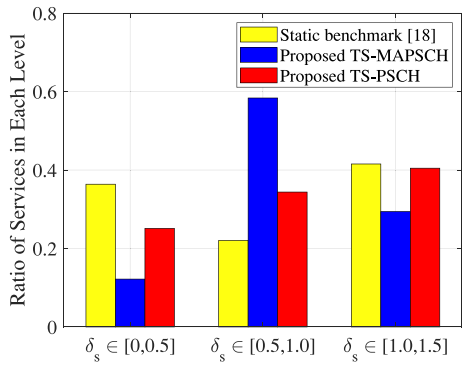

Fig. 5. Comparison of QoS satisfaction level of all the accepted services between the static benchmark algorithm and the proposed TS-based algorithms in the low-migration-cost scenario.

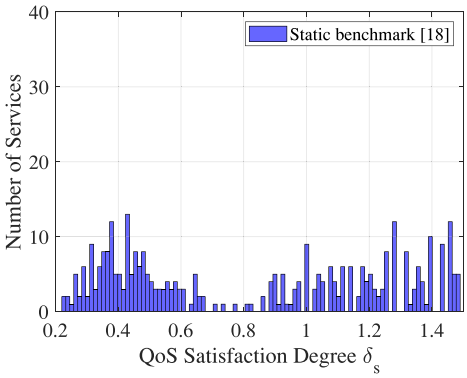

Fig. 6. Histogram of QoS satisfaction degree (δ_s) for the accepted services using the static benchmark algorithm.

proposed TS-MAPSCH algorithm is more advantageous than the other three algorithms in comparison.

2) QoS Satisfaction Level: Next, we show the advantage of the proposed algorithms in terms of increasing the ability of the substrate network to satisfy services' QoS requirements. We define another performance metric called QoS satisfaction degree as the ratio of a service request's actual E2E delay (i.e., the E2E delay experienced by a service when the processing of its last VNF is finished) over its delay requirement. Let the QoS satisfaction degree for service s be denoted by $\delta_s = (t^c_s - t^a_s)/D_s$. For all the three algorithms, we calculate the QoS satisfaction degree for each accepted service and group them into three levels, i.e., [0, 0.5], [0.5, 1.0], and [1.0, 1.5]. The ratio of the accepted services in each group is shown in Fig. 5. As can be shown in the figure, with the static benchmark algorithm, the ratios of the services in the three groups are about 40%, 20%, and 40%, respectively. By using the proposed TS-MAPSCH algorithm, these ratios are adjusted to be 15%, 60%, and 25%, respectively. This shows that the proposed dynamic algorithm increases the total profit by averaging the QoS satisfaction degree of the accommodated services. The histograms of QoS satisfaction degree for the accepted services for the static benchmark and the proposed TS-MAPSCH algorithm are shown in Figs. 6 and 7, respectively. It can be observed from Fig. 7 that using TS-MAPSCH,

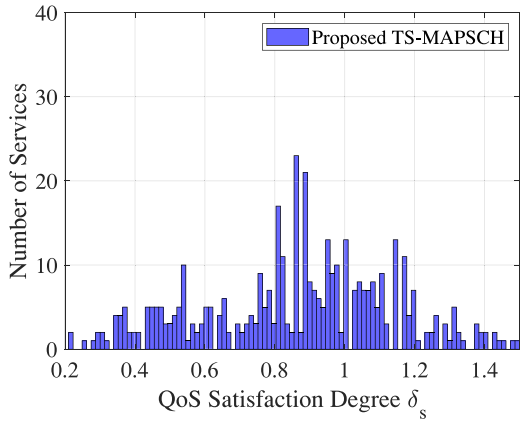

Fig. 7. Histogram of QoS satisfaction degree (δ_s) for the accepted services using the proposed TS-MAPSCH algorithm.

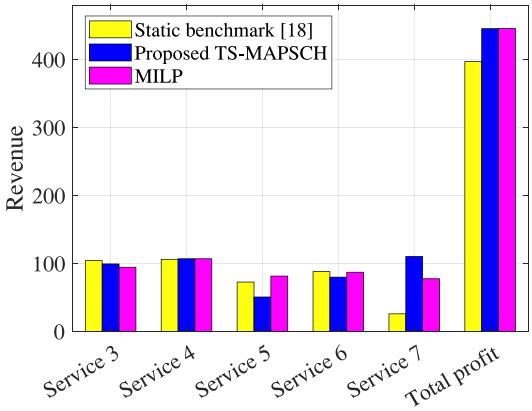

Fig. 8. Performance comparison of total profit between static benchmark, the proposed TS-MAPSCH algorithm and MILP.

most of the accepted services are with a QoS satisfaction degree in the range of [0.8, 1.2].

3) Performance Gap Between the Proposed Solution and the Optimal Solution From MILP: To show the performance gap between the solution from the proposed TS-based algorithms and the optimal solution, we solve the MILP model established at a time instant when the proposed TS-based algorithm (i.e., TS-MAPSCH or TS-PSCH) was triggered in the low-migration-cost online scenario described previously. Specifically, the current system time under consideration is chosen to be $t_c = 45$. At this time instant, service request s_7 newly arrives to the system (i.e., $\bar{s} = s_7$) and its delay requirement cannot be strictly satisfied in the initial mapping and scheduling strategy obtained by using the greedy algorithm. If we use the static algorithm to accommodate s_7, without performing any dynamic operations, the total profit for all the unfinished services will be 397.14. The proposed TS-based algorithm is then triggered with five unfinished services to be remapped and rescheduled, i.e., $\mathcal{S}_u = \{s_3, s_4, s_5, s_6, s_7\}$. Using the proposed TS-MAPSCH algorithm, the total profit for the unfinished services in $\mathcal{S}_u$ at t_c is increased from 397.14 to 445.27, while guaranteeing that none of their deadlines is violated. For comparison purposes, we capture the network status at t_c and feed all the important parameters as shown in Table I as inputs to the MILP to find the optimal solution. The optimal solution from solving MILP results in an increase in the total profit for $\mathcal{S}_u$ from 397.1 to 445.58. This demonstrates that the performance gap between the proposed TS-MAPSCH algorithm and MILP is very small and acceptable. Fig. 8 shows more detailed results in terms of each unfinished service's actual revenue after the adjustment and the total profit. It can be seen that by using the proposed algorithm, the total profit is very close to the optimal solution.

4) Running Time Comparison: We further compare the running time for offline optimization among the three algorithms. All the three algorithms are run on a Legion Y540 Laptop with an Intel i7-9750H CPU @2.6 GHz. Table IV compares the running time as well as the total profit for the three algorithms. We can find that solving the MILP model using Gurobi solver to find the optimal solution takes about 43 s. In comparison, using the proposed TS-MAPSCH algorithm to obtain

TABLE IV
RUNNING TIME AND TOTAL PROFIT COMPARISONS

Metric	Static benchmark [18]	MILP	TS-MAPSCH
Running time	$<$ 1 ms	43.15 s	0.226 s
Total profit	397.14	445.58	445.27

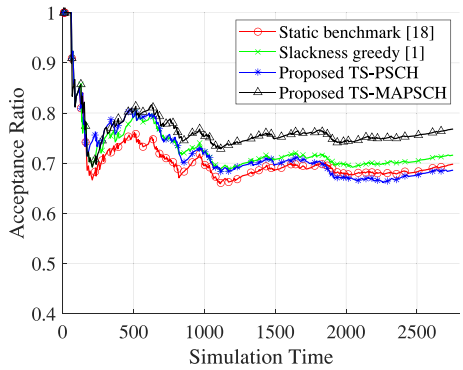

Fig. 9. Comparison of acceptance ratio between the two existing algorithms and the proposed TS-based algorithms in the high-migration-cost scenario.

near-optimal solutions only takes 0.226 s, which is significantly faster than solving MILP. On the other hand, the time for running the static benchmark algorithm is less than 1 ms, but its total profit is (more than 10%) less than that of the other two algorithms. In summary, the proposed TS-MAPSCH algorithm achieves more total profit than the static benchmark algorithm with acceptable additional time complexity.

B. High-Migration-Cost Online Scenario

Next, the performance of the four algorithms is compared in a high-migration-cost scenario, where $p_{sf}^m = 8.0$ and $p_{sf}^r = 4.0$. The simulation is carried out with 500 services and the service arrival rate is set to be 0.2 service per time unit. The maximum E2E delay violation degree is set as $\epsilon_{\max} = 0.5$. The comparison between the four algorithms in terms of service acceptance ratio and the total profit is shown in Figs. 9 and 10, respectively. It can be observed from Fig. 9 that the advantage of the two proposed TS-based algorithms in terms of increasing the acceptance ratio becomes smaller as the migration cost becomes higher. The reason is that when the migration cost is high, the gain on the profit is not as considerable as that in the

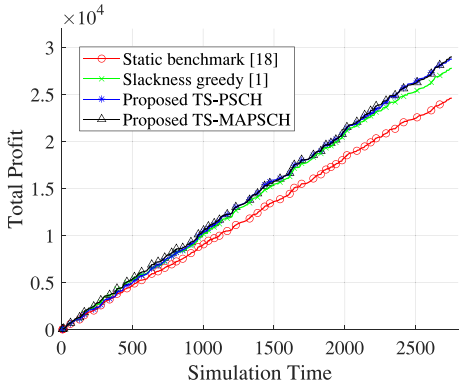

Fig. 10. Comparison of total profit between the two existing algorithms and the proposed TS-based algorithms in the high-migration-cost scenario.

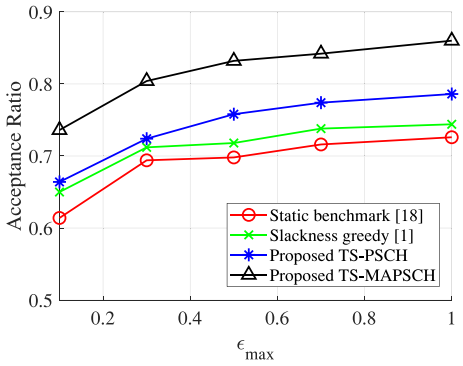

Fig. 11. Comparison of acceptance ratio versus maximum allowed E2E delay violation ($\epsilon_{\max}$) between the two existing algorithms and the proposed TS-based algorithms in the low-operational-cost scenario.

low-migration-cost scenario. It is not uncommon that the new mapping and scheduling strategy found by the TS procedure does not deserve to be executed, as the total profit may even decrease. Consequently, the new service will be mapped and scheduled directly using the GEP algorithm, leading to a lower acceptance ratio. The performance of the proposed TS-PSCH algorithm in terms of acceptance ratio is quite close to that of the two existing algorithms, while the proposed TS-MAPSCH achieves about 10% higher acceptance ratio than the two existing algorithms. As for the total profit, we can see from Fig. 10 that the two proposed TS-based algorithms and the slackness greedy algorithm perform equally well and all better than the static algorithm. At the end of the simulation, the total profit achieved by the two proposed algorithms is about 20% higher than that achieved by the static benchmark. This demonstrates that even in high-migration-cost scenarios, the proposed TS-based algorithms are still able to increase the service provider's total profit in the long run.

C. Performance Comparison for Varied E2E Delay Violation Tolerance ϵ_{max}

Figs. 11 and 12 show the comparisons among the four algorithms with varied E2E delay violation tolerance (i.e., $\epsilon_{\max}$ is chosen from [0.1, 0.3, 0.5, 0.7, 1.0]) in the low-operational-cost scenario (i.e., $p^m_{sf} = 0.4$ and $p^r_{sf} = 0.2$). It can be seen from Fig. 11 that as the delay tolerance varies from 0.1 to 1.0, the service acceptance ratio achieved by the

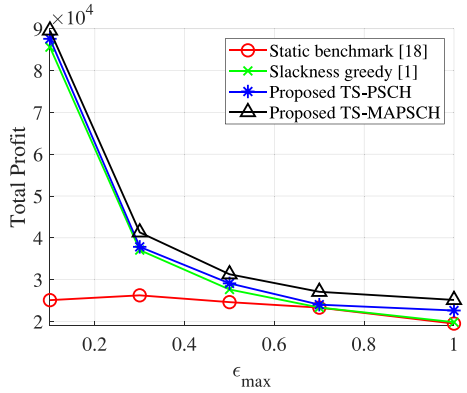

Fig. 12. Comparison of total profit versus maximum allowed E2E delay violation ($\epsilon_{\max}$) between the two existing algorithms and the proposed TS-based algorithms in the low-operational-cost scenario.

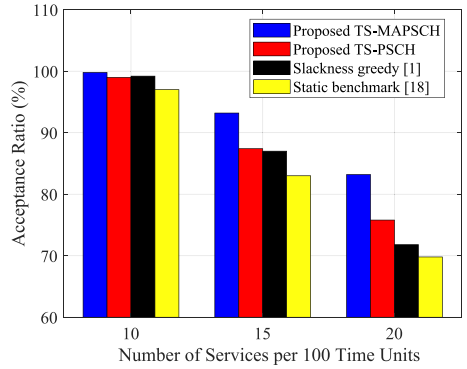

Fig. 13. Comparison of acceptance ratio versus arrival rate between the two existing algorithms and the proposed TS-based algorithms in the low-operational-cost scenario.

proposed TS-based algorithms is consistently higher than that of the two existing algorithms. In addition, as $\epsilon_{\max}$ increases, the number of service requests accommodated by the SAGIN increases gradually for all the four algorithms in comparison. This is reasonable as a larger $\epsilon_{\max}$ implies that a larger delay violation can be tolerated, and thus leads to a higher acceptance ratio. As for the total profit, as shown in Fig. 12, the proposed algorithms constantly outperform the two existing algorithms as $\epsilon_{\max}$ varies from 0.1 to 1.0. However, the performance gap between the proposed TS-based algorithms and the static benchmark decreases as $\epsilon_{\max}$ increases. This indicates that despite the acceptance ratio improvement brought by VNF remapping and rescheduling, the total profit enhancement diminishes if the delay requirements of existing services are severely violated.

D. Performance Comparison for Varied Service Arrival Rates

Figs. 13 and 14 show the comparison among the four algorithms with varied service arrival rates (chosen from [0.1, 0.15, 0.2]) in the low-operational-cost scenario ($p^m_{sf} = 0.4$ and $p^r_{sf} = 0.2$) in terms of the acceptance ratio and total profit, respectively. The maximum allowed E2E delay violation is set as $\epsilon_{\max} = 0.5$. It can be seen from Figs. 13 and 14 that for all the three simulation settings, the proposed TS-based algorithms outperform the two existing algorithms

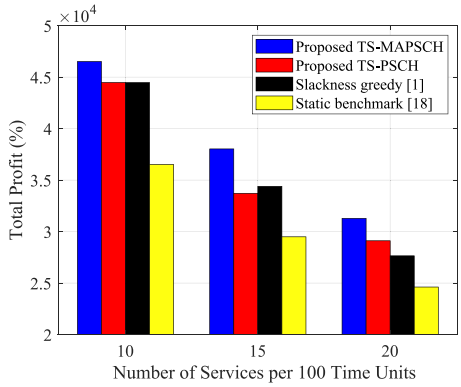

Fig. 14. Comparison of total profit versus arrival rate between the two existing algorithms and the proposed TS-based algorithms in the low-operational-cost scenario.

in terms of acceptance ratio and total profit. This is due to the fact that our proposed dynamic algorithms allow remapping and/or rescheduling of the existing VNFs in the network to increase the chance of admitting newly arrived services. At the same time, VNF remapping and rescheduling also optimizes the allocation and utilization of the physical resources. In addition, we can also observe that the performance gap between the proposed TS-based algorithms and the two existing algorithms is enlarged as service arrival rate increases. With a small service arrival rate, physical resources are adequate to accommodate the newly arrived services. Therefore, when the network load is light, the room for further performance improvement is limited for the proposed remapping and rescheduling strategies.

VII. Conclusion

In this article, dynamic VNF mapping and scheduling has been jointly considered to achieve QoS-guaranteed IoV service provisioning in the SAGIN. Two TS-based heuristic algorithms have been designed to obtain suboptimal solutions to the proposed MILP problem, which allow the existing VNFs in the system to be migrated, reinstantiated, or rescheduled efficiently. Extensive simulation results have been provided to demonstrate the effectiveness of the proposed algorithms in improving the service acceptance ratio and total profit. The online VNF remapping and rescheduling studied in this work provides a theoretical basis for studies related to SDN/NFV-based service provisioning in the highly dynamic IoV scenarios. In addition, the principle of considering network heterogeneity in the VNF mapping and scheduling strategy design can be valuable for SFC deployments in heterogeneous radio access networks. In future works, we will analyze and incorporate the link capacity constraint and radio link transmission latency into the problem formulation. We will also consider the stability and quality of internodes wireless links (e.g., LEO-to-BS, UAV-to-LEO, inter-UAV, etc.) in the design of the dynamic VNF mapping and scheduling algorithms.

References

[1] J. Li, W. Shi, P. Yang, and X. Shen, "On dynamic mapping and scheduling of service function chains in SDN/NFV-enabled networks," in *Proc. IEEE Glob. Commun. Conf. GLOBECOM*, Dec. 2019, pp. 1–6.

[2] J. Contreras-Castillo, S. Zeadally, and J. A. Guerrero-Ibañez, "Internet of vehicles: Architecture, protocols, and security," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3701–3709, Oct. 2018.

[3] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[4] J. Liu, Y. Shi, Z. Md Fadlullah, and N. Kato, "Space-air-ground integrated network: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2714–2741, 4th Quart., 2018.

[5] S. Zhang, W. Quan, J. Li, W. Shi, P. Yang, and X. Shen, "Air-ground integrated vehicular network slicing with content pushing and caching," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2114–2127, Sep. 2018.

[6] S. Zhou, G. Wang, S. Zhang, Z. Niu, and X. Shen, "Bidirectional mission offloading for agile space-air-ground integrated networks," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 38–45, Apr. 2019.

[7] I. F. Akyildiz and A. Kak, "The Internet of space Things/CubeSats: A ubiquitous cyber-physical system for the connected world," *Comput. Netw.*, vol. 150, pp. 134–149, Feb. 2019.

[8] G. Araniti, G. Genovese, A. Iera, A. Molinaro, and S. Pizzi, "Virtualizing nanosatellites in SDN/NFV enabled ground segments to enhance service orchestration," in *Proc. IEEE Glob. Commun. Conf. GLOBECOM*, Dec. 2019, pp. 1–6.

[9] Z. Zhou, J. Feng, C. Zhang, Z. Chang, Y. Zhang, and K. M. S. Huq, "SAGECELL: Software-defined space-air-ground integrated moving cells," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 92–99, Aug. 2018.

[10] 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, Jul. 2020.

[11] A. J. Gonzalez, G. Nencioni, A. Kamisinski, B. E. Helvik, and P. E. Heegaard, "Dependability of the NFV orchestrator: State of the art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3307–3329, 4th Quart., 2018.

[12] J. G. Herrera, and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[13] M. A. Ferrag, L. Maglaras, and A. Ahmim, "Privacy-preserving schemes for ad hoc social networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 3015–3045, 4th Quart., 2017.

[14] O. Alhussein *et al.*, "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, Jun. 2020.

[15] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF placement and CPU allocation in 5G," in *Proc. IEEE Conf. Comput. Commun. INFOCOM*, Apr. 2018, pp. 1943–1951.

[16] M. A. T. Nejad, S. Parsaeefard, M. Maddah-Ali, T. Mahmoodi, and B. H. Khalaj, "vSPACE: VNF simultaneous placement, admission control and embedding" *IEEE J. Sel. Areas Commun.* vol. 36, no. 3, pp. 542–557, Mar. 2018.

[17] F. Wang, R. Ling, J. Zhu, and D. Li, "Bandwidth guaranteed virtual network function placement and scaling in datacenter," in *Proc. IEEE 34th Int. Perform. Comput. Commun. Conf. IPCCC*, Dec. 2015, pp. 1–8.

[18] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Proc. IEEE Netw. Softwarization (NetSoft)*, Apr. 2015, pp. 1–9.

[19] J. F. Riera, E. Escalona, J. Batallé, E. Grasa, and J. A. García-Espín, "Virtual network function scheduling: Concept and challenges," in *Proc. Int. Conf. Smart Commun. Netw. Technol. (SaCoNeT)*, Jun. 2014, pp. 1–5.

[20] J. Li, W. Shi, N. Zhang, and X. Shen, "Delay-aware VNF scheduling: A reinforcement learning approach with variable action set," *IEEE Trans. Cogn. Commun. Netw.*, early access, Apr. 21, 2020, doi: 10.1109/TCCN.2020.2988908.

[21] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Commun.*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.

[22] H. A. Alameddine, L. Qu, and C. Assi, "Scheduling service function chains for ultra-low latency network services," in *Proc. 13th Int. Conf. Netw. Service Manag. (CNSM)*, Nov. 2017, pp. 1–9.

[23] C. Pham, N. H. Tran, and C. S. Hong, "Virtual network function scheduling: A matching game approach," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 69–72, Jan. 2018.

[24] L. Guo, J. Pang, and A. Walid, "Dynamic service function chaining in SDN-enabled networks with middleboxes," in *Proc. IEEE 24th Int. Conf. Netw. Protocols ICNP*, Nov. 2016, pp. 1–10.

[25] H. Hawilo, M. Jammal, and A. Shami, "Orchestrating network function virtualization platform: Migration or re-instantiation?" in *Proc. Cloud Netw. (CloudNet)*, Sep. 2017, pp. 35–40.
[26] D. Cho, J. Taheri, A. Y. Zomaya, and P. Bouvry, "Real-time virtual network function (vnf) migration toward low network latency in cloud environments," in *Proc. IEEE Cloud Comput. (CLOUD)*, Jun. 2017, pp. 798–801.
[27] J. Zhang, L. Li, and D. Wang, "Optimizing VNF live migration via para-virtualization driver and QuickAssist technology," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[28] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2008–2025, Aug. 2017.
[29] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 3, pp. 543–553, Sep. 2017.
[30] N. Kato *et al.*, "Optimizing space-air-ground integrated networks by artificial intelligence," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 140–147, Aug. 2019.
[31] X. Li, W. Feng, Y. Chen, C.-X. Wang, and N. Ge, "Maritime coverage enhancement using UAVs coordinated with hybrid satellite-terrestrial networks," *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2355–2369, Apr. 2020.
[32] H. Wu, F. Lyu, C. Zhou, J. Chen, L. Wang, and X. Shen, "Optimal UAV caching and trajectory in aerial-assisted vehicular networks: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 12, pp. 2783–2797, Dec. 2020.
[33] W. Shi, J. Li, H. Wu, C. Zhou, N. Cheng, and X. Shen, "Drone-cell trajectory planning and resource allocation for highly mobile networks: A hierarchical DRL approach," *IEEE Internet Things J.*, early access, Apr. 28, 2020, doi: 10.1109/JIOT.2020.3020067.
[34] Y. Ruan, Y. Li, C.-X. Wang, R. Zhang, and H. Zhang, "Power allocation in cognitive satellite-vehicular networks from energy-spectral efficiency tradeoff perspective," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 2, pp. 318–329, Jun. 2019.
[35] S. R. Pokhrel, J. Jin, and H. L. Vu, "Mobility-aware multipath communication for unmanned aerial surveillance systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6088–6098, Jun. 2019.
[36] J. C. Bennett and H. Zhang, "WF^2Q: Worst-case fair weighted fair queueing," in *Proc. IEEE Conf. Comput. Commun. INFOCOM*, vol. 96, Mar. 1996, pp. 120–128.
[37] Q. Ye, J. Li, K. Qu, W. Zhuang, X. Shen, and X. Li, "End-to-end quality of service in 5G networks: Examining the effectiveness of a network slicing framework," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 65–74, Jun. 2018.
[38] M. Gendreau, and J. Y. Potvin, "Tabu search," in *Search Methodologies*. Boston, MA, USA: Springer, 2005, pp. 165–186.

Junling Li (Graduate Student Member, IEEE) received the B.S. degree from Tianjin University, Tianjin, China, in 2013, the M.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2016, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2020.

She is currently a Joint Postdoctoral Research Fellow with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Chinese University of Hong Kong, Shenzhen, China, and the University of Waterloo. Her research interests include game theory, machine learning, software-defined networking, network function virtualization, and vehicular networks.

Dr. Li received the Best Paper Award at the IEEE/CIC International Conference on Communications in China in 2019.

Weisen Shi (Graduate Student Member, IEEE) received the B.S. degree from Tianjin University, Tianjin, China, in 2013, the M.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2016, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2020.

His research interests include space–air–ground-integrated networks, UAV communication and networking, and RAN slicing.

Huaqing Wu (Student Member, IEEE) received the B.E. and M.E. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 2014 and 2017, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada.

Her current research interests include vehicular networks with emphasis on edge caching, resource allocation, and space–air–ground-integrated networks.

Shan Zhang (Member, IEEE) received the Ph.D. degree in electronic engineering from Tsinghua University, Beijing, China, in 2016.

She is currently an Associate Professor with the School of Computer Science and Engineering, Beihang University, Beijing. She was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, from 2016 to 2017. Her research interests include mobile edge caching, wireless network virtualization, and intelligent management.

Dr. Zhang received the Best Paper Award at the Asia–Pacific Conference on Communication in 2013.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular *ad hoc* and sensor networks.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) presents in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society, and Technical Recognition Award from Wireless Communications Technical Committee in 2019 and AHSN Technical Committee in 2013, and the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'7, the Symposia Chair for the IEEE ICC'10, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the IEEE Communications Society President Electronics. He was the Vice President for Technical and Educational Activities, Vice President for Publications, Member-at-Large on the Board of Governors, Chair of the Distinguished Lecturer Selection Committee, and Member of IEEE Fellow Selection Committee. He served as the Editor-in-Chief for the IEEE INTERNET OF THINGS JOURNAL, IEEE NETWORK, and *IET Communications*. He is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.