Received 5 September 2022; revised 18 October 2022; accepted 7 November 2022. Date of publication 10 November 2022; date of current version 21 November 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3221168

Service Deployment Model on Shared Virtual Network Functions With Flow Partition

JINGXIONG ZHANG (Graduate Student Member, IEEE), FUJUN HE (Member, IEEE), AND EIJI OKI (Fellow, IEEE)

Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan

CORRESPONDING AUTHOR: J. ZHANG (e-mail: zhang.jingxiong.58e@st.kyoto-u.ac.jp)

This work was supported in part by the Japan Society for the Promotion of Science KAKENHI, Japan, under Grant 21H03426.
A part of this article was presented in [1] at IEEE International Conference on Communications, May 2022 [DOI: 10.1109/ICC45855.2022.9838826].

ABSTRACT Network operators can operate services in a flexible way with virtual network functions thanks to the network function virtualization technology. Flow partition allows aggregated traffic to be split into multiple parts, which increases the flexibility. This paper proposes a service deployment model with flow partition to minimize the service deployment cost with meeting service delay requirements. A virtual network function of a service is allowed to have several instances, each of which hosts a part of flows and can be shared among different services, to reduce the initial and proportional cost. We provide the mathematical formulation for the proposed model and transform it to a special case as a mixed integer second-order cone programming (MISOCP) problem. A heuristic algorithm, which is called a flow partition heuristic (FPH), is introduced to solve the original problem in practical time by decomposing it into several steps; each step handles a convex problem. We compare the performances of proposed model with flow partition and conventional model without flow partition. We consider the formulated MISOCP problem with adopting a strategy of even splitting to divide flows in a special case, which is called an even spitting heuristic (ESH). The performances of FPH and ESH are compared in a realistic scenario. We also consider the formulated MISOCP problem as an original problem and compare it to an FPH-based heuristic algorithm with the even-splitting strategy (FPH-ES), in both realistic and synthetic scenarios. The numerical results reveal that the proposed model saves the service deployment cost compared to the conventional one. It improves the maximum admissible traffic scale by 23% in average in our examined cases. We observe that FPH outperforms ESH and ESH outperforms FPH-ES in terms of the service deployment cost in their own focused problems, respectively.

INDEX TERMS Network function virtualization, service deployment, flow partition, queueing theory.

I. INTRODUCTION

NETWORK functions (NFs), such as firewalls and load balancers, are able to be implemented in a flexible way to share the infrastructure resources with the help of network function virtualization (NFV) technology. By decoupling NFs from their dedicated physical network equipments, a given service can be decomposed into a set of virtual network functions (VNFs) which can be deployed on commodity servers [2], [3]. Given a set of services that consist of requested VNFs, a critical problem is how to deploy these VNFs, including placing VNFs and allocating computing resources, with meeting the requirements of services.

The service deployment problem, or the deployment problem of VNFs that belong to the services, is widely studied in recent works [4], [5], [6], [7], [8], [9], [10], [11], [12]. Usually, given a set of virtual machines (VMs), a set of requested VNFs, and a set of coming services, the service deployment problem decides which VM to host which VNF assigned to which service to optimize an objective, such as to minimize the required computing capacity, with satisfying several constraints, such as the service delay constraint.

In literature, such as the above works, for each VNF type required by a service function chain (SFC), one VNF instance (VNFI) is usually assigned to handle all flows of the SFC. Considering the capacity constraint on each VNFI,

the number of feasible combinations of SFCs that share the same VNFI is limited. Therefore, this approach is not flexible and efficient in terms of resource utilization. Intuitively, for each VNF type, if the flows of an SFC can be partitioned into different sets, each of which uses one of the multiple assigned VNFIs or replicas, SFCs can combine more freely. In other words, the degree of VNF sharing among SFCs can be improved; network resources can be provisioned more flexibly. We introduce this flexible approach as *flow partition*.

The technology to enable flow partition has been studied from different aspects. On one hand, the computing capacity of a virtual middlebox can be split with maintaining the same functionality, based on the works to develop multiple VNFIs or replicas. Clearly, a stateless virtual middlebox, such as a packet compression, can be easily split into a set of replicas to process flows independently [13]. For a stateful virtual middlebox, such as an intrusion detection system, the work in [14] introduced a system called FreeFlow to support elastic and correct execution among replicas. FreeFlow identifies and divides a virtual middlebox's states into two classes: internal and external. The internal state is only required by a single replica; the external state is shared among replicas. Through ensuring that each replica can access the states, internal and/or external, required to produce the appropriate outputs for the incoming traffic, the consistency for traffic processing is achieved. In other words, multiple replicas of one VNF can exist in a network to work in parallel; traffic for this VNF can be distributed among replicas. On the other hand, the software defined networking (SDN) technology provides flexible control and data planes that enable us to partition the flows of an SFC and to transmit each flow to its VNFI through an appropriate path. Several works utilized the SDN paradigm with multipath routing to achieve a high performance transmission, such as to improve the reliability with the diversity coding [15] or to perform load balancing [16].

As the previous works focused on either the frameworks to run multiple VNFIs or the traffic routing schemes with multipath, there is no study addressing a model to deploy SFCs with considering flow partition. Such a model needs to answer how much the ratio of each partition of flows is, which VNFI is assigned to each partition of flows, where to place each VNFI, and how much capacity needs to be allocated to a component. This is the focus of this work.

This paper proposes a service deployment model with flow partition to minimize the service deployment cost with meeting the delay requirement of services. We set the maximum allowed number of partitions for each VNF in each service. We provide the mathematical formulation for the proposed model and transform it to a special case as a mixed integer second-order cone programming (MISOCP) problem. In order to solve the problem in practical time, a flow partition heuristic (FPH) is introduced to decompose the problem into several steps; each step handles a convex problem. We consider the formulated MISOCP problem with adopting an even-splitting strategy to divide flows in a special case as a heuristic of the original problem, which is called an even-splitting heuristic (ESH), and compare it to FPH in a realistic scenario. We then consider the formulated MISOCP problem as the original problem and an FPH-based heuristic algorithm with the even-slitting strategy called FPH-ES is introduced. The performances of ESH and FPH-ES are compared in both realistic and synthetic scenarios. The numerical results show that compared to the conventional model, the proposed model saves the service deployment cost. It improves the maximum admissible traffic scale by 23% in average in our examined cases. The results also reveal that FPH outperforms ESH and ESH outperforms FPH-ES in terms of the service deployment cost in their own focused problems, respectively.

This paper is an extended version of [1] with various additions, which are mainly described as follows. We extensively survey existing studies related to service deployment in NFV. We refine the heuristic introduced in [1], and analyze its time complexity; the refined heuristic is called FPH in this paper. We introduce ESH and FPH-ES as heuristics in addition to FPH. We conduct more experiments on different aspects to further evaluate the proposed model by using these heuristics with the dependency of service deployment cost and the number of activated VMs on traffic scale. We also evaluate the heuristics with computation time and the dependency of maximum admissible number of divisions. We analyze the obtained numerical results to show the advantage of flow partition and how flexibility of flow partition influences the performance of model.

The rest of the paper is organized as follows. Section II introduces related works. Section III describes the motivation and background of this work. The problem is then formulated and the mathematical transformation for a special case is introduced. Section IV presents the heuristic algorithm for the formulated problem. Section V shows numerical results to evaluate the proposed model. Section VI concludes this paper.

II. RELATED WORK

The service deployment problem is considered from different aspects, each of which has their own objectives and constraints. Table 1 shows the summary of works related to service deployment problems. The work in [4] developed a model to minimize the energy consumption, which is defined as the summation of allocated computing resources and the consumption of activated links and nodes. It considers the robustness level of resource demand uncertainty and latency constraints on SFCs, as well as the tradeoff between these points. The work in [5] introduced a double deep Q network (DDQN) model to deploy VNFs with the help of deep reinforcement learning technology. Learning from information about network devices, traffic and resources, an optimal policy considering deployment cost and the rejection of SFC request (SFCR) is generated. The work in [6] considered to minimize the overall delay including inter-cloud and link

TABLE 1. Summary of works on service deployment problem.

Reference	Objective	Service delay	Link capacity (bandwidth)	Uncertainty	VNF decomposition	Queueing model	Traffic splitting
[4]	Minimizing energy consumption	✓	✓	✓(demand)			
[5]	Minimizing VNF deployment cost and penalty of reject SFC requests	✓	✓	✓(load)			
[6]	Minimizing inter-cloud and link queueing latency	✓	✓			✓	
[7]	Minimizing average delay of multiple classes of services	✓	✓		✓	✓	
[8]	Minimizing maximum link utilization and VNF deployment cost	✓	✓				
[9]	Minimizing link bandwidth utilization	✓	✓		✓		✓
[10]	Minimizing total delay	✓				✓	
[11]	Minimizing service deployment cost	✓				✓	
[12]	Minimizing service deployment cost	✓		✓(traffic)		✓	
[18]	Minimizing node and link cost of substrate network		✓		✓		
[20]	Minimizing maximum link utilization		✓	✓(demand)			✓
This work	Minimizing service deployment cost	✓			✓	✓	✓

queueing delay with satisfying the link capacity and delay requirements. The work in [7] jointly considered VNF placement, resource assignment, and traffic routing for vertical services. The work in [8] developed a model to minimize the maximum of link utilization as well as the allocated computing resources. It assumes that multiple types of VNFs can be deployed on one VM, but the flows of one VNF are not allowed to be divided. It also defines multiple VM templates, where the required cost of computing resources and latency performance for one VNF are different in each VM template. Compared to the work in [8], our work allows the flows to be divided into more than one part and deployed on different VMs. Our model uses an M/M/1 queueing system to estimate the processing delay of VMs depending on the allocated computing capacity and arrival rate of traffic flows.

Several works considered VNF decomposition to improve the flexibility of resource utilization [9], [17], [18]. The work in [17] introduced the evolved packet core (EPC) user-plane function that is decomposed into serving gateway (SGW) and packet data network gateway (PGW) sub-functions by classifying them in meta function groups. The common function groups within different sub-functions can reduce the cost by eliminating dedicated hardware and unleashing the function placement restrictions. Function decomposition is also utilized to analyze the components of EPC network in a fine-grained level to find what type of functions can be merged to improve the network structure. The work in [18] decomposed one VNF into multiple sub-functions for virtual network slice (vNS) so that the resources of the same type of sub-functions in different vNS requests can be shared. Interconnections among different VNFs are considered; the node resource consumptions are reduced and the total cost of substrate network is minimized. The work in [9] introduced a service provisioning model with considering VNF decomposition, where two types of VNF decomposition were introduced. The first type is to decompose one function into several sub-functions, each of which realizes a part of functionality of the original function. In other words, a flow needs to visit all the sub-functions for one functionality. In the second type of decomposition, each sub-function provides the whole functionality with reduced computing capacity. It indicates that a flow can visit one of the sub-functions for one functionality, which is similar to the idea of the split/merge system presented in [14] and flow partition in this paper. The work in [9] formulated a mixed integer linear programming (MILP) problem for VNF deployment with considering diversified VNF decomposition and hybrid multipath routing, which is based on the resource-constrained assignment problem. The VNF sharing among different services is not considered in [9]. In addition, the queueing and processing delays are assumed as a per-hop delay, which is given in [9]; our work estimates them, which depend on the result of resource allocation, based on queueing systems.

Similar to our work, several studies considered the service deployment problems based on the queueing systems to be aware of service delay more explicitly [6], [10], [11], [12]. The work in [10] incorporated the Poisson distribution of packet arrival rate and packet size in an M/M/1 system to deal with non-uniform distribution of traffic flows. It focused on the limited processing capacity of the NFV servers and end-to-end delay including queueing delay in each server and

link delay. The work in [11] adopted different approaches of traffic priority for VNF sharing. The model introduced in [11] considered the process of running a VNF on a VM as an M/M/1 system with priority queueing. The work in [12] solved the VNF deployment problem based on first-in-first-out queueing. The model presented in [12] took traffic uncertainty into consideration, where the traffic arrival rate is not always deterministic in practical applications. Our work defines the processing time of VNF belonging to the service as well as the total processing time of a service based on an M/M/1 queueing system.

Our work is also related to traffic splitting, which has been widely studied for load distribution over multipath networks. Typically, the traffic splitting can be classified based on the level of splitting granularity, which mainly includes packet-level, flow-level, and subflow-level [19]. The work in [20] developed a multipath routing algorithm to solve the problem of minimizing the maximum link utilization. The algorithm proposed in [20] is able to adapt to dynamic change of service demands and improve the throughput of links. The work in [21] implemented a multipath transmission framework with introducing a traffic splitting approach, where a flow is split into several flow units with various sizes in the source node based on NFV. The idea of flow partition is based on flow-level splitting to partition flows of a service such that different flows can use different VNFIs and paths, instead of splitting packets of a flow into sub-flows. The work in [20], [21] focused on the transmission aspect, i.e., the multipath routing, for traffic splitting; the processing aspect, i.e., the VNF deployment for services, was not considered. Different from [20], [21], this work focuses on the VNF deployment and computing resource allocation with flow partition.

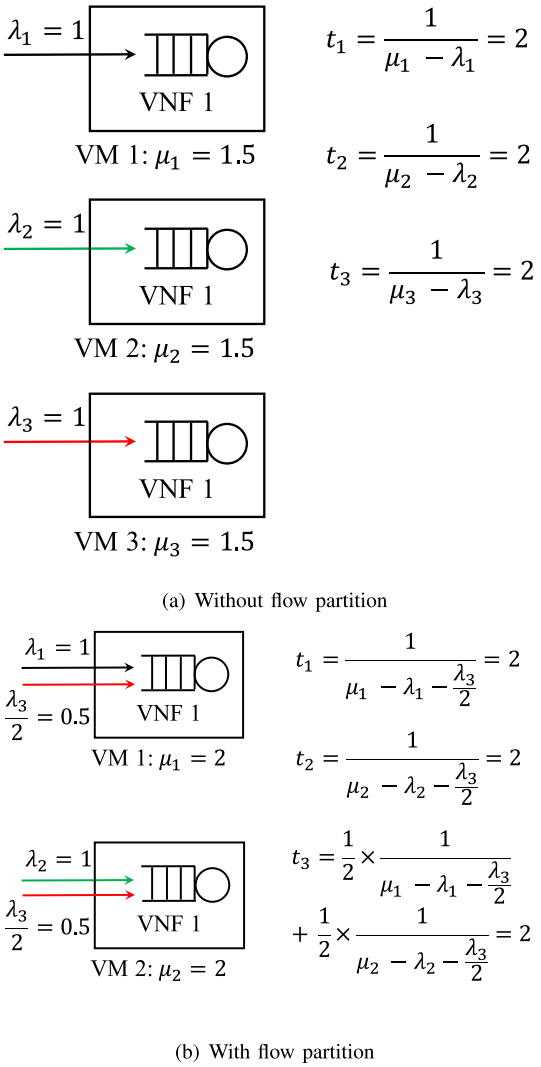


FIGURE 1. Example of VNF deployments with and without flow partition.

III. MODEL AND PROBLEM FORMULATION

A. MOTIVATION

We present an example to show our motivation to consider the flow partition. Consider three VMs, each of which has the maximum processing rate of 2. There are three SFCs, each of which only requires VNF 1 with the traffic arrival rate of 1, where the maximum admissible delay is 2. The initial cost to activate a VM and the proportional cost to allocate each unit of computing capacity to a VM are considered as 2 and 1, respectively.

Figure 1(a) shows the VNF deployment with the minimum total cost required for the conventional approach, which does not consider the flow partition. To guarantee the system stability that the service rate needs to be greater than the traffic arrival rate at each VM, a dedicated VNFI of VNF 1 is deployed in a VM for each SFC. To satisfy the maximum admissible delay for each SFC, the minimum capacity allocated in each VM is 1.5 such that the delay is $\frac{1}{1.5-1} = 2$, as shown in Fig. 1(a); the delay is expressed by $\frac{1}{\mu-\lambda}$ in an M/M/1 system with the traffic arrival rate of λ and the service rate, i.e., allocated capacity, of μ. We observe that it requires the total initial cost of $2 \times 3 = 6$ to activate the three VMs, where each VM is allocated with the capacity of 1.5, which leads to the total proportional cost of $(1 \times 1.5) \times 3 = 4.5$. As a result, the total cost of $6 + 4.5 = 10.5$ is required for the conventional approach.

Figure 1(b) shows the VNF deployment with the minimum required total cost when the flow partition is considered, where two VNFIs are deployed. The flows of SFC 3 are equally partitioned into two parts, each of which goes to each VNFI. In other words, a flow of SFC 3 has the probability of $\frac{1}{2}$ to visit each of the two VNFIs. To satisfy the maximum admissible delay for each SFC, the minimum capacity allocated in each VM is 2 such that the delay for flows at each VM is $\frac{1}{2-(1+0.5)} = 2$, as shown in Fig. 1(b). We observe that the total initial cost and the total proportional cost are $2 \times 2 = 4$ and $(1 \times 2) \times 2 = 4$, respectively, which leads to the total cost of $4 + 4 = 8$. Compared to the conventional approach, $10.5 - 8 = 2.5$ is saved for the total cost when the flow partition is considered; both initial cost and proportional cost are reduced.

B. QUEUEING SYSTEMS WITH FLOW PARTITION

1) QUEUEING SYSTEM

Consider a set of services and a set of VNFs, which are denoted by S and V, respectively. For each $s \in S$, the set of VNFs is a non-empty subset of V. Let g_{sv} denote a given binary parameter; it equals one if service s uses VNF $v \in V$, and zero otherwise. A set of VMs, which is denoted by M, is used to run VNFs. In the model, considering reducing the complexity of VM operation for network operators, it is supposed that each VM $m \in M$ can run at most one VNF $v \in V$. Let x^s_{vm} be a binary variable, which equals 1 if VNF v belonging to service s runs on VM m and 0 otherwise. The running process on each VM $m \in M$ is considered as an M/M/1 system using queueing theory. The computing capacity of VM m is denoted by $a_m \in [0, C_m]$, where C_m denotes the maximum value of computing capacity that VM m can be scaled up. Let l_v be the required computing capacity of VNF v to process a flow with unit time; the service rate of VM m, μ_m, is expressed as $\frac{a_m}{l_v}$. λ_{sv} denotes the arrival rate of traffic flows belonging to service s for VNF v, and $\lambda_{sv} = 0$ means that service s does not include VNF v. The maximum delay of service s is denoted by $D^{\max}_s$; processing time of VNF v belonging to service s, which is denoted by t_{sv}, can be expressed as $t_{sv} = \frac{1}{\frac{a_m}{l_v} - \lambda_{sv} - \sum_{s' \in S \setminus \{s\}} \lambda_{s'v} x^{s'}_{vm}}$, where VNF v is deployed on VM m. Total processing time of service s, which is denoted by T_s, is calculated by adding up all the processing time of VNFs belonging to it: $T_s = \sum_{v \in V} t_{sv} g_{sv}$.

2) FLOW PARTITION

Consider that flows of service $s \in S$ for VNF $v \in V$ can be divided into at most d_{sv} parts, where d_{sv} denotes a given positive integer. Let y^i_{sv}, $i \in [1, d_{sv}]$, $s \in S$, $v \in V$, represent a binary variable; it is set to one if there exists the ith part of flows, and zero otherwise. Let k^i_{sv}, $i \in [1, d_{sv}]$, $s \in S$, $v \in V$, represent the proportion of ith part of flows. We have:

$$\sum_{i=1}^{d_{sv}} k^i_{sv} y^i_{sv} = g_{sv}, \qquad \forall s \in S, v \in V, \tag{1}$$

$$0 \le k^i_{sv} \le 1, \qquad \forall i \in [1, d_{sv}], s \in S, v \in V. \tag{2}$$

Equation (1) indicates that the sum of proportions equals one if service s uses VNF v. Equation (2) shows the range of each proportion. For y^i_{sv}, we give the following constraints:

$$y^1_{sv} = g_{sv}, \qquad \forall s \in S, v \in V, \tag{3}$$

$$y^i_{sv} \le y^{i-1}_{sv}, \qquad \forall i \in [2, d_{sv}], s \in S, v \in V. \tag{4}$$

Equation (3) ensures that the first part of flows always exists if service s includes VNF v. Equation (4) indicates that the ith part of flows can exist only when the $(i-1)$th part of flows exists. Let z^m_{svi} be a binary variable, which equals one if the ith part of flows is deployed on VM m, and zero otherwise. We have:

$$\sum_{m \in M} z^m_{svi} = y^i_{sv}, \qquad \forall i \in [1, d_{sv}], s \in S, v \in V. \tag{5}$$

Equation (5) expresses that each part of flows is deployed on one VM if this part of flows exists. Let w_{mv} denote a binary variable; it is set to one if VM $m \in M$ hosts a VNFI for VNF $v \in V$, and zero otherwise. We consider that one VM can deploy at most one VNF [11], [22], which is expressed as:

$$\sum_{v \in V} w_{mv} \le 1, \qquad \forall m \in M. \tag{6}$$

Note that the idea of flow partition can be extended to the case that one VM can deploy more than one VNF. All parts of flows are deployed on different VMs that host the VNFI for VNF v. It is expressed as:

$$\sum_{i=1}^{d_{sv}} z^m_{svi} \le w_{mv}, \qquad \forall s \in S, v \in V, m \in M. \tag{7}$$

The system stability constraint is given as:

$$\sum_{s \in S} \sum_{v \in V} \sum_{i=1}^{d_{sv}} k^i_{sv} \lambda_{sv} z^m_{svi} l_v < a_m, \qquad \forall m \in M. \tag{8}$$

Equation (8) indicates that the total arrival rate of flows on VM m does not reach its allocated service rate. Time required on VNF v including the processing time and the waiting time for service s is expressed as:

$$t_{sv} = \sum_{i=1}^{d_{sv}} k^i_{sv} \sum_{m \in M} z^m_{svi} \frac{l_v}{a_m - \sum_{s' \in S} \sum_{v' \in V} \sum_{i'=1}^{d_{s'v'}} k^{i'}_{s'v'} \lambda_{s'v'} z^m_{s'v'i'} l_{v'}}, \quad \forall s \in S, v \in V. \tag{9}$$

3) PROBLEM FORMULATION

VNFs belonging to services need to be deployed such that all services are finished in the maximum admissible average delay. We form an optimization problem to illustrate the service deployment model considering flow partition. The optimization problem rigorously defines the objective and constraints. The optimization problem can be a reference to inspire heuristic algorithms and mathematical transformation so that the problem can be solved in practical time. We focus on the capital expenditure of the network operator when implementing flow partition, i.e., the utilization of computing capacity resources, in this work.

Table 2 summarizes the notations frequently used in this paper. Let K^{f}_m be the initial cost to activate VM $m \in M$ and K^{u}_m be the cost of computing capacity for each unit in VM m.

We formulate the flow partition problem to minimize the total required deployment cost as the following optimization problem:

TABLE 2. List of notations frequently used in this paper.

	Notations	Descriptions
Given parameters	S	Set of services
	V	Set of VNFs
	M	Set of VMs
	C_m	Maximum computing capacity of VM $m \in M$
	K_m^{f}	Cost of activating VM $m \in M$
	K_m^{u}	Cost of computing capacity for each unit in VM $m \in M$
	l_v	Units of computing capacity to achieve the processing rate of one flow per unit time for VNF $v \in V$
	λ_{sv}	Arrival rate of traffic flows belonging to service $s \in S$ for VNF $v \in V$
	g_{sv}	Binary. 1 if service s includes VNF $v \in V$, and 0 otherwise
	d_{sv}	Integer. Maximum admissible number of parts that flows of service $s \in S$ for VNF $v \in V$ can be divided
	$D_s^{\max}$	Maximum admissible delay of service $s \in S$
	$\mathcal{K}_{sv}^{ij}$	Proportion of the ith part of flows of service $s \in S$ for VNF $v \in V$ when divided into j parts.
Variables	a_m	Allocated computing capacity of VM $m \in M$
	t_{sv}	Processing time of service $s \in S$ for VNF $v \in V$
	T_s	Processing time of service $s \in S$
	y_{sv}^i	Binary. 1 if there exists the ith part of flows of service $s \in S$ for VNF $v \in V$, and 0 otherwise
	k_{sv}^i	Proportion of the ith part of flows of service $s \in S$ for VNF $v \in V$
	x_{vm}^s	Binary. 1 if VNF $v \in V$ of service $s \in S$ is deployed on VM $m \in M$, and 0 otherwise
	z_{svi}^m	Binary. 1 if the ith part of flows of service $s \in S$ for VNF $v \in V$ is deployed on VM $m \in M$, and 0 otherwise
	w_{mv}	Binary. 1 if VM $m \in M$ hosts a VNFI for VNF $v \in V$, and 0 otherwise
	b_{sv}^j	Binary. 1 if the flows of service $s \in S$ for VNF $v \in V$ are divided into j parts, and 0 otherwise

$$\min \quad \sum_{m\in M}\left(K_m^{\mathrm{f}}\sum_{v\in V} w_{mv} + K_m^{\mathrm{u}} a_m\right) \tag{10a}$$

$$\text{s.t.} \quad (1)-(9) \tag{10b}$$

$$a_m \le C_m \sum_{v\in V} w_{mv}, \forall m \in M \tag{10c}$$

$$\sum_{v\in V} t_{sv} \le D_s^{\max}, \forall s \in S \tag{10d}$$

$$y_{sv}^i \in \{0,1\}, \forall i \in [1, d_{sv}], s \in S, v \in V \tag{10e}$$

$$z_{svi}^m \in \{0,1\}, \forall i \in [1, d_{sv}], s \in S, v \in V, m \in M \tag{10f}$$

$$w_{mv} \in \{0,1\}, \forall v \in V, m \in M. \tag{10g}$$

Equation (10a) minimizes the total cost required to deploy the VNFs. Equation (10c) shows that the allocated capacity of VM m cannot exceed its maximum capacity if it is activated; if there is no VNF deployed on it, the allocated capacity is zero. Equation (10d) means that the average processing delay of each service is within its maximum admissible average delay. Equation (10e)-(10g) indicate the ranges of decision variables y_{sv}^i, z_{svi}^m, and w_{mv}, respectively.

4) MATHEMATICAL TRANSFORMATION FOR SPECIAL CASE

Since (10d) combined with (8) and (9) contains several real variables, especially k_{sv}^i, with the fraction form, it is difficult to transform (10a)-(10g) to a formulation that can be handled by optimization solvers [23], [24]. We describe a special case that this issue is addressed. We assume that the value of k_{sv}^i is prepared by the network operator in advance given the number of partitions, or $\sum_{i=1}^{d_{sv}} y_{sv}^i$. Let $\mathcal{K}_{sv}^{ij}, i, j \in [1, d_{sv}], s \in S, v \in V$, denote a given parameter indicating the proportion of ith part of flows when the flows are divided into j parts. Let b_{sv}^j denote a binary variable; it is set to one if the flows are divided into j parts and zero otherwise. We replace k_{sv}^i with $\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij} b_{sv}^j$. We have:

$$\sum_{j=1}^{d_{sv}} b_{sv}^j = g_{sv}, \forall s \in S, v \in V \tag{11a}$$

$$\sum_{j=1}^{d_{sv}} b_{sv}^j j = \sum_{i=1}^{d_{sv}} y_{sv}^i, \forall s \in S, v \in V. \tag{11b}$$

Equation (11a) indicates that the flows of VNF v belonging to service s can only be divided by one certain j if the flows of VNF v belonging to service s exist. Equation (11b) shows that the number of divided parts of flows is in accordance with the number of exisitng parts of flows. Then, (1), (8), and (9) are transformed to:

$$\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij} b_{sv}^j y_{sv}^i = g_{sv}, \forall s \in S, v \in V \tag{12a}$$

$$\sum_{s\in S}\sum_{v\in V}\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij} b_{sv}^j \lambda_{sv} z_{svi}^m l_v < a_m, \forall m \in M \tag{12b}$$

$$t_{sv} = \sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij} b_{sv}^j \sum_{m\in M} z_{svi}^m \frac{l_v}{a_m - \sum_{s'\in S}\sum_{v'\in V}\sum_{i'=1}^{d_{s'v'}}\sum_{j'=1}^{d_{s'v'}} \mathcal{K}_{s'v'}^{i'j'} b_{s'v'}^{j'} \lambda_{s'v'} z_{s'v'i'}^m l_{v'}}, \quad \forall s \in S, v \in V. \tag{12c}$$

Equation (12a) is further transformed to:

$$\alpha_{sv}^{ij} = b_{sv}^j y_{sv}^i, \forall i, j \in [1, d_{sv}], s \in S, v \in V \tag{13a}$$

$$g_{sv} = \sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij} \alpha_{sv}^{ij}, \forall s \in S, v \in V, \tag{13b}$$

where α_{sv}^{ij} is a binary variable introduced for linearization. Equation (13a) is then linearized by:

$$\alpha_{sv}^{ij} \le b_{sv}^j, \forall i, j \in [1, d_{sv}], s \in S, v \in V \tag{14a}$$

$$\alpha_{sv}^{ij} \le y_{sv}^i, \forall i, j \in [1, d_{sv}], s \in S, v \in V \tag{14b}$$

$$\alpha_{sv}^{ij} \ge b_{sv}^j + y_{sv}^i - 1, \forall i, j \in [1, d_{sv}], s \in S, v \in V. \tag{14c}$$

Similarly, (12b) is linearized by:

$$\beta_{svm}^{ij} = b_{sv}^{j} z_{svi}^{m}, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (15a)$$

$$\sum_{s\in S}\sum_{v\in V}\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij}\lambda_{sv}l_v\beta_{svm}^{ij} < a_m, \forall m \in M \quad (15b)$$

$$\beta_{svm}^{ij} \le b_{sv}^{j}, \forall i \in [1, d_{sv}], j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (15c)$$

$$\beta_{svm}^{ij} \le z_{svi}^{m}, \forall i \in [1, d_{sv}], j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (15d)$$

$$\beta_{svm}^{ij} \ge b_{sv}^{j} + z_{svi}^{m} - 1, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M, \quad (15e)$$

where β_{svm}^{ij} is a binary variable introduced for linearization. We then consider how to deal with (12b) and (12c):

$$\delta_m = \sum_{s\in S}\sum_{v\in V}\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij}\lambda_{sv}l_v\beta_{svm}^{ij}, \forall m \in M \quad (16a)$$

$$\delta_m < a_m, \forall m \in M \quad (16b)$$

$$t_{sv} \ge \sum_{m\in M}\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} \mathcal{K}_{sv}^{ij}\beta_{svm}^{ij}\frac{l_v}{a_m - \delta_m}, \forall s \in S, v \in V. \quad (16c)$$

Equation (16a) defines δ_m, which denotes the required capacity to process traffic flows deployed on VM m. Equation (16b), which is a replacement of (12b), ensures the system stability. Equation (16c) is obtained from (12c) indicating the sojourn time of service s on VNF v.

We introduce two types of variables, e_{svm}^{ij} and v_m, for transformation. Then, (16c) is transformed to:

$$e_{svm}^{ij} = \mathcal{K}_{sv}^{ij}\beta_{svm}^{ij}\frac{l_v}{a_m - \delta_m}, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (17a)$$

$$t_{sv} \ge \sum_{m\in M}\sum_{i=1}^{d_{sv}}\sum_{j=1}^{d_{sv}} e_{svm}^{ij}, \forall s \in S, v \in V \quad (17b)$$

$$v_m = a_m - \delta_m, \forall m \in M \quad (17c)$$

$$e_{svm}^{ij} v_m \ge \mathcal{K}_{sv}^{ij}\beta_{svm}^{ij} l_v, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M. \quad (17d)$$

With introducing a real variable of p_{svm}^{ij}, we then consider how to transform (17d):

$$p_{svm}^{ij} = e_{svm}^{ij} + v_m, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (18a)$$

$$\left(p_{svm}^{ij}\right)^2 \ge \left(e_{svm}^{ij}\right)^2 + (v_m)^2 + 2\left(\sqrt{\mathcal{K}_{sv}^{ij} l_v}\beta_{svm}^{ij}\right)^2, \quad \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M \quad (18b)$$

$$p_{svm}^{ij} \ge 0, \forall i, j \in [1, d_{sv}], s \in S, v \in V, m \in M. \quad (18c)$$

By replacing (1), (8), and (9) in (10a)-(10g) with (11a), (11b), (13b), (14a)-(14c), (15c)-(15e), (16a), (16b), (17b), (17c), and (18a)-(18c), we transform the problem to an MISOCP problem in the special case, which can be handled by optimization solvers [23], [24].

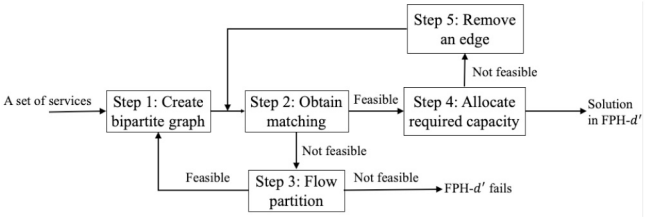


FIGURE 2. Flowchart of FPH-d'.

Algorithm 1 Flow Partition Heuristic (FPH)

Input: d
Output: T_d

```
Initialize d' = 1, T'_d = +∞
while d' ≤ d do
    Obtain T_{d'} by Algorithm 2 (FPH-d')
    T'_d = min(T'_d, T_{d'})
    d' ← d' + 1
end while
Set T_d = T'_d
return T_d
```

IV. HEURISTIC ALGORITHM

We introduce a flow partition heuristic algorithm called FPH (see Algorithm 1) to solve the problem in practical time. We assume that $d_{sv} = d, \forall s \in S, v \in V$, in the algorithm. Given d, FPH iteratively runs FPH-d' (see Algorithm 2) for each positive integer $d' \le d$, and returns the smallest obtained $T_{d'}$ as the minimum deployment cost T_d. Figure 2 outlines the running process of FPH-d' for a certain d'. After the algorithm finishes running, d' is updated as $d'+1$ and FPH-d' starts running with new d' from the initial case until d' reaches d. Figure 3 shows an example to demonstrate different steps of FPH-d'. Consider that there is a set of services requesting VNFs v_1 and v_2. Since the same VNF is shared among different services, we categorize the services requesting the same VNF into one set. Let $S_v = \{s | g_{sv} = 1, \forall s \in S\}$ denote the set of services which requires VNF $v \in V$. Let ξ_v denote the set of subsets of S_v, where the intersection of any two elements in ξ_v is empty; the union of all elements in ξ_v is S_v. Let $\xi = \bigcup_{v\in V}\xi_v$ denote the set of elements in $\xi_v, v \in V$. In the initial case, we consider that $\xi_v = \{S_v\}, \forall v \in V$, where $\xi_{v_1} = \{S_{v_1}\} = \{\{s_1, s_2, s_3\}\}$ and $\xi_{v_2} = \{S_{v_2}\} = \{\{s_2, s_3, s_4\}\}$, as shown in Fig. 3(a), and $\xi = \{\{s_1, s_2, s_3\}, \{s_2, s_3, s_4\}\}$.

A. STEPS 1 AND 2: CREATE BIPARTITE GRAPH AND OBTAIN MINIMUM WEIGHT MATCHING

At Step 1, FPH-d' creates a bipartite graph considering a set of services and VMs, where a node on one side refers to an element in ξ, and a node on the other side refers to a VM. There exists an edge if the maximum computing capacity provided by a VM is larger than the total traffic rate of an element in ξ, and the cost to deploy the requested VNF on

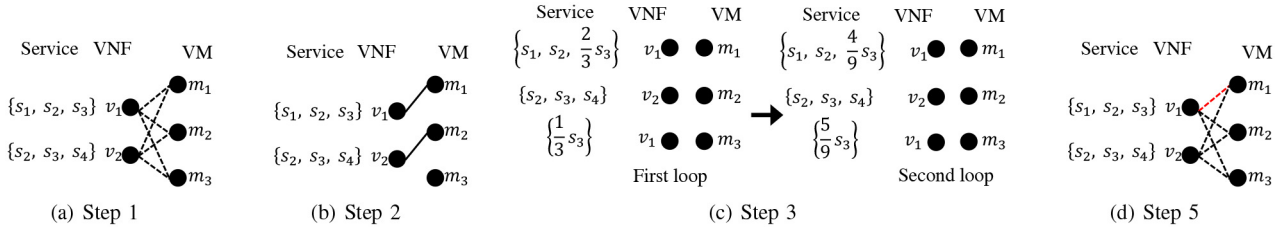


FIGURE 3. Algorithm steps, where dashed lines in black and red represent possible and removed matchings, respectively; solid lines represent obtained matchings.

Algorithm 2 FPH With Specific Maximum Number of Divisions d' (FPH-d')

Input: d'
Output: $T_{d'}$

Initialize $step = 1$
while true **do**
  if $step = 1$ **then**
    Step 1: create bipartite graph
    $step = 2$
  else if $step = 2$ **then**
    Step 2: obtain minimum weight matching
    if a feasible matching is obtained **then**
      $step = 4$
    else
      $step = 3$
    end if
  else if $step = 3$ **then**
    Step 3: flow partition (see Algorithm 3)
    if the algorithm meets the failure conditions **then**
      break
    else
      $step = 1$
    end if
  else if $step = 4$ **then**
    Step 4: allocate capacity
    if a feasible solution is obtained **then**
      return $T_{d'}$
    else
      $step = 5$
    end if
  else
    Step 5: remove an edge
    $step = 2$
  end if
end while

that VM, including K_m^{f} and K_m^{u}, denotes the weight on the edge.

At Step 2, FPH-d' finds a VM for each set of VNFs to deploy by obtaining the minimum weight matching based on the bipartite graph formed in Step 1. Hungarian algorithm [25] is used to solve the matching problem in polynomial time and obtains the matching whose total cost is minimized.

Algorithm 3 Step 3 in FPH-d'

Let ξ^{old} and ξ^{new} denote set ξ before and after Step 3, respectively.
Input: ξ^{old}
Output: ξ^{new}

Step 3.1: Calculate the approximate required capacity of each set $\zeta \in \xi^{\mathrm{old}}$, and decide $\zeta^\star$ to be divided.
Step 3.2: Calculate the approximate required capacity of each service $s \in \zeta^\star$, and decide $s^\star$ to be divided.
Step 3.3: Calculate Δ.
if $\zeta^\star$ has been selected in previous loops **then**
  move Δ from $s^\star$ to the created set.
else
  move Δ from $s^\star$ to an empty set.
end if
ξ^{old} is updated as ξ^{new}.
if any of the failure condition is met **then**
  FPH-d' fails.
else
  Go to Step 1.
end if

B. STEP 3: FLOW PARTITION

If Step 2 fails, it indicates that the total arrival rates of elements in ξ are so large that the whole element of services cannot be deployed on VMs. We consider using flow partition at Step 3 (see Algorithm 3) to divide one subset of ξ_v in ξ and deploy the divided part on another VM. Step 3 consists of three sub-steps: decide which set to divide (Step 3.1); decide which service to divide (Step 3.2); and decide how to divide the selected service for the selected VNF (Step 3.3).

1) STEP 3.1: DECIDE WHICH SET TO DIVIDE

We first decide which set to divide. In order to meet the stability constraints of VMs, we approximately calculate the required capacity of each set $\zeta \in \xi$ and divide the one which requires the most. FPH evenly distributes the maximum admissible delay of service s on each involved VNF; the admissible delay of each VNF belonging to service s is expressed as: $\frac{D_s^{\max}}{\sum_{v\in V} g_{sv}}$. Let k_s^ζ denote the proportion of VNF belonging to service s in set ζ. In the initial case, we set $k_s^\zeta = 1, \forall s \in \zeta, \zeta \in \xi$. Therefore, the maximum admissible delay of element ζ is expressed as: $\min_{s\in\zeta} \frac{D_s^{\max}}{\sum_{v\in V} g_{sv}}$, and the approximate required capacity is expressed as:

$l_v(\frac{1}{\min_{s\in\xi}\frac{D_s^{\max}}{\sum_{v\in V} g_{sv}}} + \sum_{s\in\zeta} k_s^{\zeta}\lambda_{sv})$. Let U_ζ, which is a given parameter, denote the number of times that element $\zeta \in \xi$ can be selected. The element that can be selected, or $U_\zeta > 0$, and is the largest approximate required capacity is selected, which is denoted as $\zeta^\star$. Let $v^\star$ denote the type of VNF in set $\zeta^\star$. We divide $\zeta^\star$ into two sets by placing the divided part from the original set into another set.

2) STEP 3.2: DECIDE WHICH SERVICE TO DIVIDE

Then we decide the VNF of which service in $\zeta^\star$ is divided. Similarly, we estimate the required capacity of the VNF of service s in $\zeta^\star$, which is expressed as: $l_{v^\star}(\frac{1}{\frac{D_s^{\max}}{\sum_{v\in V} g_{sv}}} + k_s^{\zeta^\star}\lambda_{sv^\star})$. The service with the largest approximated value, which is denoted as $s^\star$, is selected and divided in order to meet the stability constraint of VMs, if the number of partitions for this service is not larger than its maximum admissible one, or $d_{s^\star v^\star}$; we select the largest one that is feasible following a decreasing order of the approximated values, otherwise. If it is the first time that $\zeta^\star$ is selected and all the services in it reach their own maximum admissible number of division, FPH-d' puts the whole flows of service $s^\star$ for VNF $v^\star$ into a new set and activate a new VM to deploy it. This ensures that FPH-d' can handle all possible $d_{sv^\star}, \forall s \in \zeta^\star$.

3) STEP 3.3: DECIDE HOW TO DEVIDE SELECTED SERVICE FOR SELECTED VNF

Then we consider how to divide the VNF $v^\star$ belonging to service $s^\star$. Let Δ denote the amount of flows that the algorithm moves from $s^\star$ to another set. We intend to set the value of Δ small enough to make sure that the VNF belonging to service $s^\star$ can be divided successfully in each loop; the precision of division is taken into consideration to ensure that the remaining capacity of VMs is used sufficiently. On the other hand, the number of required loops can increase as the the value of Δ decreases. Here, we set $\Delta = \min_{s\in\zeta^\star}\frac{k_s^{\zeta^\star}\lambda_{sv^\star}}{d_{sv^\star}}$. After each time of division at Step 3, the value of $k_{s^\star}^{\zeta^\star}$ is updated, $U_{\zeta^\star}$ is decreased by one, and FPH-d' goes back to Step 1. Considering improving the usage efficiency of VMs, we define that in the new loop, if the selected set $\zeta^\star$ in ξ is the same as the one in previous loops, FPH-d' does not create another empty set for it but use the set created for the divided part in previous loops. As shown in Fig. 3(c), after failing at Step 2, FPH-d' enters Step 3. Let Δ_n denote the value of Δ in the nth loop.

4) EXAMPLE

We use an example to show the process of flow partition at Step 3, as depicted in Fig. 3(c). In the first loop, the element $\xi_{v_1} = \{s_1, s_2, s_3\}$ is selected, and then VNF v_1 belonging to service s_3 is selected and $\Delta_1 = \frac{1}{3}\lambda_{s_3v_1}$, where we assume $d_{s_3v_1} = 3$. Therefore, the divided part is moved from ζ_{v_1} to an empty set, and ζ_{v_1} is updated as $\{\zeta_{v_1}, \zeta'_{v_1}\} = \{\{s_1, s_2, \frac{2}{3}s_3\}, \{\frac{1}{3}s_3\}\}$, where $k_{s_3}^{\zeta_{v_1}} = \frac{\lambda_{s_3v_1}-\Delta_1}{\lambda_{s_3v_1}} = \frac{2}{3}$, and $k_{s_3}^{\zeta_{v'_1}} = \frac{\Delta_1}{\lambda_{s_3v_1}} = \frac{1}{3}$. Then FPH-$d'$ starts from Step 1. Suppose that FPH-d' still does not obtain a feasible matching, which indicates that it enters Step 3 for the second time. In the second loop, the element $\zeta_{v_1} = \{s_1, s_2, \frac{2}{3}s_3\}$ is selected, and then VNF v_1 belonging to service s_3 is selected and $\Delta_2 = \frac{2}{9}\lambda_{s_3v_1}$. Therefore, the divided part is moved from ζ_{v_1} to ζ'_{v_1}, and ζ_{v_1} is updated as: $\{\zeta_{v_1}, \zeta'_{v_1}\} = \{\{s_1, s_2, \frac{4}{9}s_3\}, \{\frac{5}{9}s_3\}\}$, where $k_{s_3}^{\zeta_{v_1}} = \frac{\frac{2}{3}\lambda_{s_3v_1}-\Delta_2}{\lambda_{s_3v_1}} = \frac{4}{9}$, and $k_{s_3}^{\zeta'_{v_1}} = \frac{\frac{1}{3}\lambda_{s_3v_1}+\Delta_2}{\lambda_{s_3v_1}} = \frac{5}{9}$.

5) FAILURE CONDITION

There are some conditions that FPH-d' is not able to obtain feasible solutions: if $U_\zeta = 0, \forall\zeta \in \xi$, and there is still no feasible matching obtained at Step 2; if the number of VMs is not enough to deploy all the parts of flows; if the traffic scale is so large that the capacity of a whole VM is not enough to deploy one part of flows while the number of parts reaches d', FPH-d' fails.

C. STEPS 4 AND 5: ALLOCATE CAPACITY AND REMOVE AN EDGE

If Step 2 obtains the minimum weight matching for all the elements in ξ, FPH determines the allocated capacity for each VM in Step 4 by solving (10a)-(10g). Since the matching and the partition are given in Step 2, the problem of (10a)-(10g) becomes a second-order cone programming (SOCP) problem, which is convex and is able to be handled by optimization solvers [23], [24].

If Step 4 fails, it indicates that the allocated capacity for VMs is not enough to satisfy the delay requirements of services, or the stability constraints of some VMs in (10c) are not satisfied due to the deployment of services on matched VMs. At Step 5, we select the VM whose remaining capacity is minimum before the deployment of services on matched VMs. Then, we remove the corresponding edge between the selected VM and the element in ξ which is deployed on it; FPH-d' restarts at Step 2 with a modified bipartite graph. As shown in Fig. 3(d), suppose that FPH-d' obtains the minimum weight matching of the bipartite graph in Fig. 3(a), but it cannot obtain a feasible solution at Step 4. The edge between VM m_1 and element $\{s_1, s_2, s_3\}$ is removed, and then FPH goes back to Step 2.

D. COMPUTATIONAL TIME COMPLEXITY OF FPH

We show that FPH is with a polynomial computational time complexity. It is obvious that FPH runs FPH-d' for d times. In FPH-d', the element in ξ can be divided for $O(\sum_{s\in S}\sum_{v\in V} d_{sv})$ times, which indicates that FPH runs Steps 1, 2, and 3 for $O(\sum_{s\in S}\sum_{v\in V} d_{sv})$ times. At each time, since there are $O(|M|\sum_{s\in S}\sum_{v\in V} d_{sv})$ edges in the bipartite graph, FPH runs Steps 4 and 5 for $O(|M|\sum_{s\in S}\sum_{v\in V} d_{sv})$ times. Then, we analyze the time complexity of each step. Step 1 makes the sets in $O(|S||V|)$ and compares the capacity between each set and VMs in $O(|M|\sum_{s\in S}\sum_{v\in V} d_{sv})$ to create the bipartite graph, which means that Step 1 needs the

time complexity of $O(|S||V| + |M| \sum_{s\in S} \sum_{v\in V} d_{sv})$. Step 2 obtains the minimum weight matching by Hungarian algorithm in $O(|M|^3)$. Step 3 determines the VNF to divide in $O(\sum_{s\in S} \sum_{v\in V} d_{sv})$, the service to divide in $O(|S|)$, and the amount of flows to divide in $O(|S|)$. As a result, Step 3 runs in $O(\sum_{s\in S} \sum_{v\in V} d_{sv} + 2|S|)$. Step 4 allocates capacity to VMs by solving the convex problem (10a)-(10g) that includes $O(|S| + |V||M|)$ constraints, and it can be solved in polynomial time. Step 5 finds the VM with minimum remaining capacity, which requires $O(\sum_{s\in S} \sum_{v\in V} d_{sv})$ times. Therefore, the computational time complexity of FPH is polynomial.

V. NUMERICAL RESULTS

We conduct several experiments to evaluate and compare the performances of conventional model, which does not consider any flow partition, and proposed model as well as its heuristics. The SOCP problems in the conventional model, the proposed model, and its modified version, and the MISOCP problem of the realistic and synthetic scenario in Section III-B4 are solved by IBM ILOG CPLEX with version 20.1 [23], running on AMD EPYC Rome 7502P, 32-core CPU, 128 GB memory.

A. EXPERIMENT SETTINGS

We consider a realistic scenario as introduced in [11]. In the realistic scenario, we use five services that are included in the smart-city domains [26], [27], [28]: intersection collision avoidance (ICA) that vehicles broadcast related information to avoid the collision, vehicular see-through (CT) that vehicles display the captured video on their on-board screens, urban sensing based on the Internet-of-Things (IoT), smart robots that are controlled through the network in a factory, and entertainment provided by streaming contents. 17 VNFs are requested in total and some of them are shared among services; the delay constraint of each service is set to 1[s]. The coefficient of each VNF, or l_v, $v \in V$, is set to 1. Table 3 shows the VNFs and their arrival rates in each service considered in the realistic scenario. We consider 45 VMs; the initial cost, the proportional cost, and the maximum capacity of each VM are set to 450, 1, and 450, respectively. We set a traffic multiplier n to regulate the scale of traffic rates. We set maximum admissible number of divisions $d_{sv} = d = 4, \forall s \in S, v \in V$, for all the experiments to investigate the dependency of deployment cost on traffic multiplier, n, which is sufficiently large so that it does not restrict the division of flows.

We also design a synthetic scenario based on the realistic scenario, in which the scale is relatively small, to obtain the optimal solution (with the relative optimality gap of 0.001 [23]) of the MISOCP problem formulated in Section III-B4. We consider three services and three VNFs in total in the synthetic scenario. The arrival rate of flows of each VNF belonging to each service is listed in Table 5, where 0 means that there is no such VNF included in the service. We consider 15 VMs; the initial cost, the proportional cost, and the maximum capacity of each VM are set to 450, 1, and 300, respectively. Other settings are the same as in the realistic scenario.

TABLE 3. VNFs and arrival rates in realistic scenario.

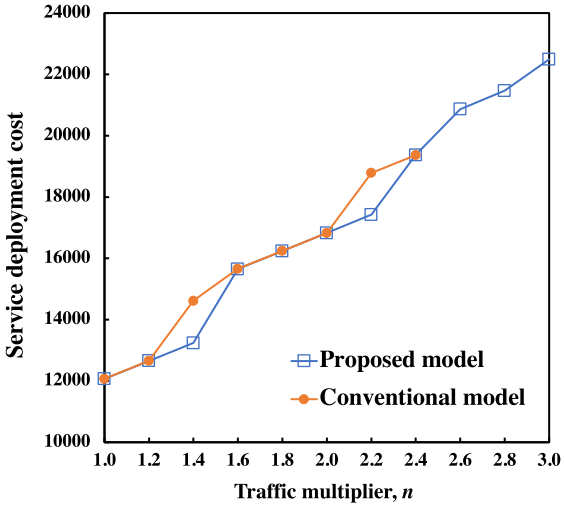
VNF	arrival rate [flows/s]
Intersection collision avoidance (ICA)	
Evolved Node B	117.69
EPC PGW	117.69
EPC SGW	117.69
EPC home subscriber server	11.77
EPC mobility management entity	11.77
Car information management (CIM)	117.69
Collision detector	117.69
Car manufacture database	117.69
Alarm generator	11.77
See through (CT)	
Evolved Node B	179.82
EPC PGW	179.82
EPC SGW	179.82
EPC home subscriber server	17.98
EPC mobility management entity	17.98
Car information management (CIM)	179.82
CT server	179.82
CT database	17.98
Sensing (IoT)	
Evolved Node B	50
EPC PGW	50
EPC SGW	50
EPC home subscriber server	5
EPC mobility management entity	5
IoT authentication	20
IoT application server	20
Smart factory (SF)	
Evolved Node B	50
EPC PGW	50
EPC SGW	50
EPC home subscriber server	5
EPC mobility management entity	5
Robotics control	50
Video feed from robots	5
Entertainment (EN)	
Evolved Node B	179.82
EPC PGW	179.82
EPC SGW	179.82
EPC home subscriber server	17.98
EPC mobility management entity	17.98
Video origin server	17.90
Video content delivery network	179.82

TABLE 4. Computation time [s] for results shown in Fig. 4.

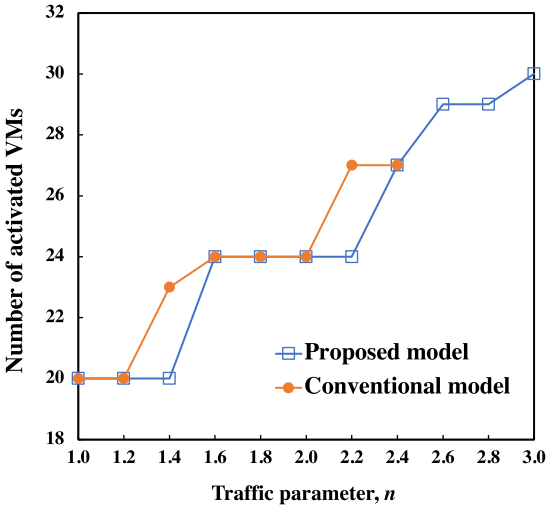
n	1.0	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0
Proposed	2.11	3.15	3.89	4.25	5.27	12.06	9.13	17.93	11.42	12.73	15.11
Conventional	0.11	0.20	0.26	0.27	0.27	0.27	0.54	0.56	—	—	—

TABLE 5. Service settings of arrival rate [flows/s] in synthetic scenario.

	VNF 1	VNF 2	VNF 3
Service 1	117.69	0	117.69
Service 2	0	50	50
Service 3	179.82	179.82	0



(a) Service deployment cost



(b) Number of activated VMs

FIGURE 4. Comparison of service deployment cost and number of activated VMs between proposed and conventional models of different values of n in realistic scenario.

B. COMPARISON BETWEEN PROPOSED AND CONVENTIONAL MODELS

We compare the proposed model with the conventional model that does not consider flow partition. The two models are handled by the introduced heuristic, FPH, and its modified version, respectively. In the modified version, we do not divide the flows of VNFI that belongs to the selected service at Step 3, but move the whole VNFI from the selected set to another set, or $k_s^\zeta = 1, \forall \zeta \in \xi, s \in S$. Figure 4 presents the service deployment cost and the number of activated VMs in the proposed and conventional models of different values of multiplier n in the realistic scenario. Figure 4(a) reveals that the service deployment cost increases as the value of n increases in both models. This is because more capacity of VMs is allocated and more VMs are activated as the traffic rates of flows are getting larger. When n is 1.4 and 2.2, the proposed model saves 9.4% and 7.3% of the deployment cost compared to the conventional model. In other cases, the deployment cost is comparable in two models. Figure 4(b) shows the same tendency as Fig. 4(a), where the proposed model needs to activate 13% and 11% less VMs than that of the conventional model when n is 1.4 and 2.2, and the number of activated VMs is the same, otherwise. This is because when the required capacity to deploy the set of VNFIs of the same VNF belonging to different services exceeds the maximum capacity of the VM, the proposed model moves parts of flows of the VNFI belonging to the selected service to another VM instead of moving the whole VNFI to another VM in the conventional model, which makes use of remaining capacity of the original VM to a larger extent. As a result, when the number of current activated VMs is the same in both models, the remaining capacity of each activated VM that deploys the same VNF can be accumulated to one VM in the proposed model. If the remaining capacity of that VM is enough to deploy the whole set of VNFIs of newly considered services, which needs to activate another VM to deploy in the conventional model, the deployment cost of activating new VMs is saved. Consequently, it saves the initial cost by activating fewer new VMs compared to the conventional model, which also explains the reason why the service deployment cost is less in the proposed model when n is 1.4 and 2.2, as shown in Fig. 4(a). It also saves the proportional cost by activating less VMs to deploy services. The proposed model saves 20 and 10 units of proportional cost compared to the conventional model when n is 1.4 and 2.2, respectively. It is noticed that when n is 2.6, 2.8 and 3.0, the conventional model does not obtain a feasible solution in these cases. This is because when the value of traffic multiplier n is larger than 2.6, the arrival rate of VNF exceeds the maximum capacity of one VM. On the other hand, the proposed model with flow partition is able to handle such large-scale traffic arrival rates and obtains feasible solutions by more flexiblely adjusting the assigned traffic for each VM.

Figure 5 observes the dependency of service deployment cost on the maximum admissible number of divisions with different traffic scales in the proposed model. We assume that $d_{sv} = d, \forall s \in S, v \in V$. In the proposed model, when $d = 1$, it is the same as the conventional model; as d is set larger, flows start to divide. Missing points in the figure indicate that there is no feasible solution obtained in those cases since the arrival rate is so large that flows for certain VNFs need to be divided into more parts to get deployed on more VMs. For all tested cases of different traffic multiplier, n, it is noticed that the deployment cost decreases or keeps the same as d becomes large. The larger admissible number

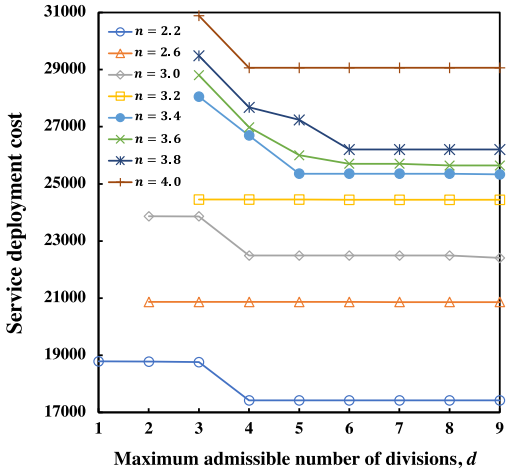


FIGURE 5. Dependency of service deployment cost on maximum admissible number of divisions, d, in proposed model.

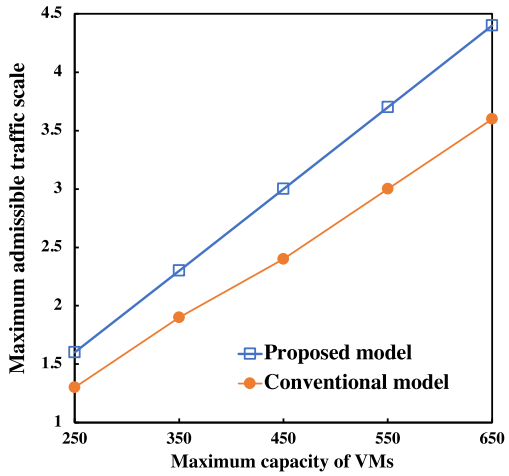


FIGURE 6. Comparison of maximum admissible traffic scale between proposed and conventional models of different values of maximum VM capacity, C_m.

of divisions improves the flexibility of division of flows and VM assignment of each part of flows to some extent. We also notice that the trend of deployment cost keeps flat when $n = 2.6$ and 3.2, and becomes flat after obvious decrease in other cases, which indicates that the flexibility of partition tends to be saturated after a certain d'.

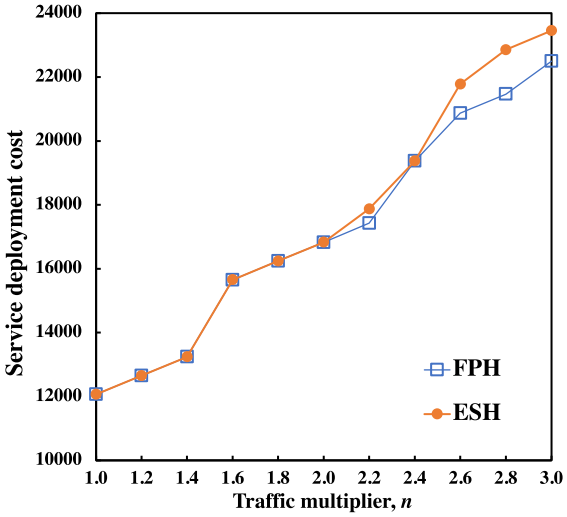
Table 4 presents the computation time to obtain the results in Fig. 4, where the value in the proposed model is the summation of all cases that $d' \leq d$; and the hyphen symbol means that there is no feasible solution obtained in those cases. As we discussed in Section IV-D, the computational time complexity of FPH, the introduced heuristic for the proposed model, is polynomial. Table 4 observes that the computation of proposed model is performed in a practical time. It shows a tendency that the computation time increases as the value of n increases in both models. This is because VNFIs of the same VNF are deployed on more VMs as the scale of flows gets larger. We notice that the computation time of the proposed model when $n = 2.0$ and 2.4 is much larger than the value around it. This is because in these cases, the FPH algorithm enters the loop at Step 5 for more times for a certain d'. We also observe that the proposed model takes 21.34 times more computation time in average than the conventional model. This is because compared to the fact that the whole VNFI is moved in one time in the conventional model, the process of moving parts of flows from the same VNFI may repeat several times to finish one time of flow partition, which consumes more time.

Figure 6 presents the maximum admissible traffic scales for different values of maximum VM capacity. The maximum admissible traffic scale is the maximum value of n that an allocation model can obtain a feasible solution given the VM capacity. We observe that it increases in both proposed and conventional models. This is because there is more available capacity for a larger arrival rate of traffic flows when the VM capacity gets larger. We also observe that the proposed model improves the maximum admissible traffic scale by 23% in average compared to the conventional model. It indicates that the proposed model is able to handle larger arrival rates of flows compared to the conventional one when the maximum capacity of VMs is the same, which further illustrates the advantage of flexibility of flow partition.

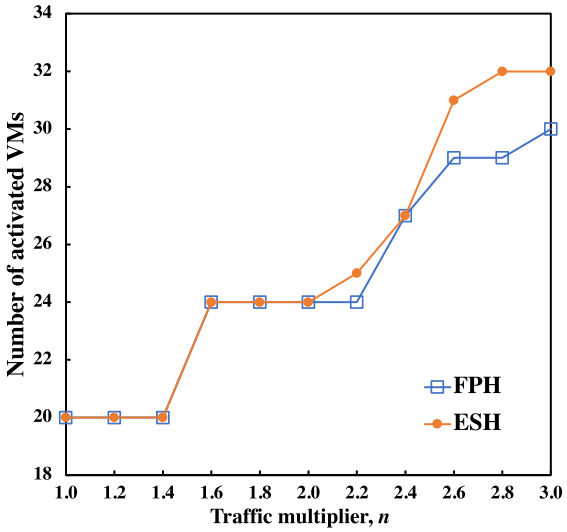
C. COMPARISON BETWEEN EVEN-SPLITTING HEURISTIC AND FPH

We consider the MISOCP approach for the special case introduced in Section III-B4 as a heuristic for the original problem, where the network operator adopts a strategy of even splitting to divide the flows, or $\mathcal{K}_{sv}^{ij} = \frac{1}{j}, \forall i, j \in [1, d_{sv}] : i \leq j$, and $0, \forall i, j \in [1, d_{sv}] : i > j, s \in S, v \in V$; we call this approach an even-splitting heuristic, called ESH. The maximum admissible computation time of ESH is set to 3600 [s] for the realistic scenario. The maximum number that VNF v belonging to service s is allowed to be divided is set to 4, or $d_{sv} = 4, \forall s \in S, v \in V$. The maximum admissible delay of each service is set to 1 [s].

Figure 7 shows the service deployment cost and number of activated VMs obtained by FPH and ESH with different multiplier n in the realistic scenario. It is shown in Figs. 7(a) and 7(b) that FPH outperforms ESH in all tested cases in terms of the service deployment cost and the number of activated VMs. This is because in ESH, the proportion of each flow can only be selected from the assignments of equally divided proportions which are decided in advance, while the FPH algorithm that can flexibly regulate each proportion of divided flows does not have such restrictions. Another reason is that the results of ESH in Fig. 7(a) are feasible solutions of the MISOCP problem obtained in limited computation time, which indicates that the value of optimal solution can be smaller. The numerical results reveal that FPH reduces the service deployment cost by 1.6% in average, and the number of activated VMs by 2.6% in average in the realistic scenario, compared to ESH.

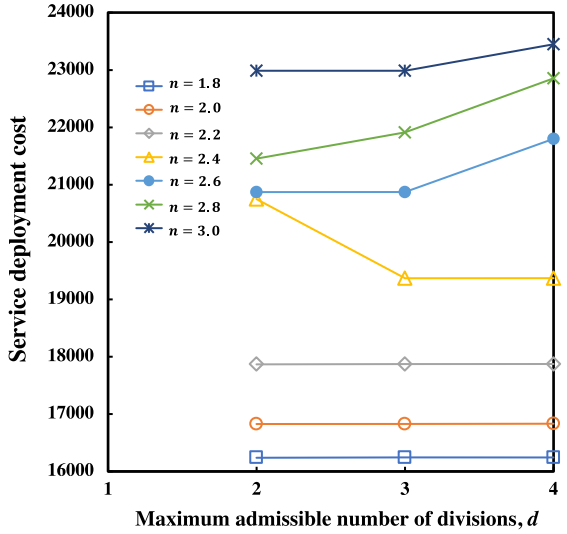


(a) Service deployment cost

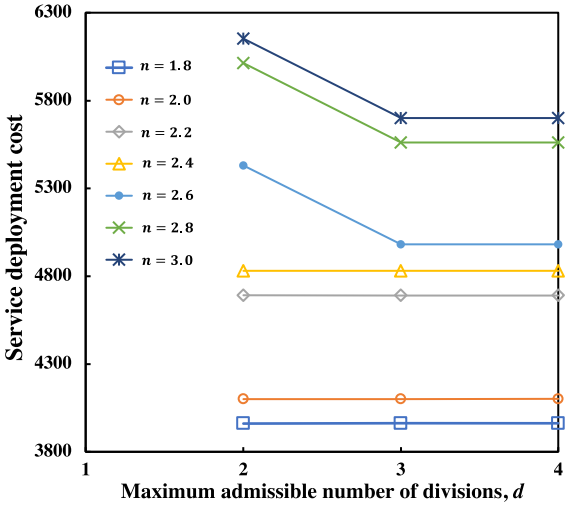


(b) Number of activated VMs

FIGURE 7. Comparison of service deployment cost and number of activated VMs between FPH and ESH of different values of n in realistic scenario.



(a) Realistic scenario

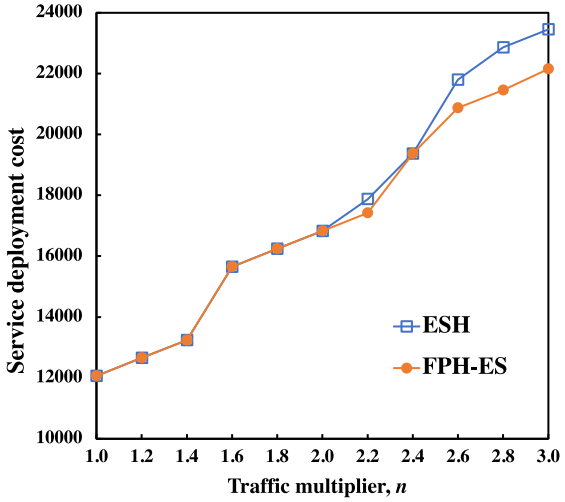


(b) Synthetic scenario

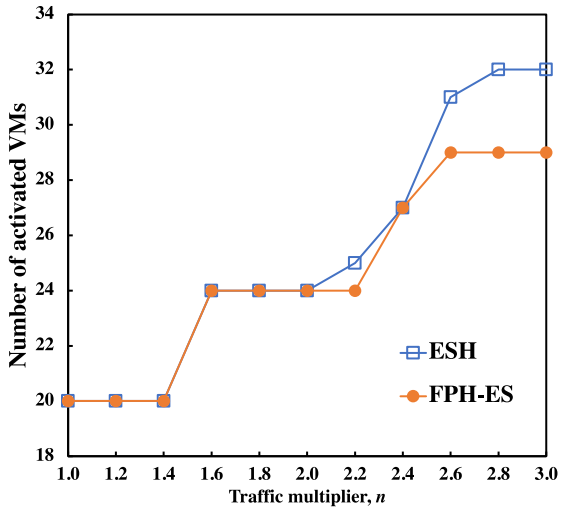
FIGURE 8. Dependency of service deployment cost on maximum admissible number of divisions, d, in ESH.

We also analyze the two heuristic algorithms, FPH and ESH, from the aspect of computation time in the realistic scenario. From Table 4, we know that the largest computation time of FPH (the proposed model with FPH) is 43.31 [s] when $n = 2.0$. On the other hand, the computation of ESH is not completed within the maximum admissible computation time of 3600 [s]. This is explained by the fact that the integer and binary variables are determined in the steps of FPH, which leads to an SOCP problem for capacity allocation; it reduces the computation time of CPLEX. In ESH, all the variables are determined in the MISOCP problem by CPLEX, which causes longer computation time even when the traffic scale is small.

Figure 8 observes the dependency of service deployment cost on the maximum admissible number of divisions with different traffic scales in ESH. We only examine the cases with $2 \leq d \leq 4$. This is because $d = 4$ is sufficient for n ranging from 1.8 to 3.0; more VMs are required if flows are divided into more parts in ESH, which indicates that the performance is not expected to be improved for $d > 4$. Figure 8(a) obtains the feasible solution in the realistic scenario within the maximum admissible computation time, which is set to 3600 [s]. We notice that as d grows larger, the performance becomes worse in some cases such as $n = 2.6, 2.8,$ and 3.0. This is because the output results are still in the process of converging to the optimal solution of the MISOCP problem before CPLEX is forced to quit due to the limitation of computation time. If the time limitation is set to a larger value, the problem in which a solution with smaller deployment cost is obtained in a larger value of d can be solved. Figure 8(b) obtains optimal solutions of the MISOCP problem in the synthetic scenario. We observe that the deployment cost decreases as d changes from 2 to 3 when $n = 2.6, 2.8,$ and 3.0 due to larger flexibility of flow



(a) Service deployment cost



(b) Number of activated VMs

FIGURE 9. Comparison of service deployment cost and number of activated VMs between ESH and FPH-ES of different values of n in realistic scenario.

partition, and keeps the same in other tested cases. This indicates that the optimal solution of ESH does not perform worse when d becomes large due to the fact that the solution space of the case with larger d includes the one with smaller d.

D. COMPARISON BETWEEN MISOCP AND INTRODUCED HEURISTIC FOR SPECIAL CASE

We consider the MISOCP problem formulated in Section III-B4 as the focused problem where a strategy of even splitting is adopted to divide the flows. We introduce an algorithm, a modified version of FPH, with adopting the even-splitting strategy for a special case to solve the problem in a shorter time; it is called FPH-ES. Similar to FPH, FPH-ES iteratively runs FPH-ES-d' for each positive integer $d' \leq d$ and returns the smallest obtained result. FPH-ES-d' is roughly the same as FPH-d' except for the fact that Step 3 of FPH-ES-d' is modified (see Algorithm 4).

Algorithm 4 Step 3 in FPH-ES-d'

Let ξ^{old} and ξ^{new} denote set ξ before and after Step 3, respectively.
Set $q_{sv} = 1, \forall s \in S, v \in V$.
Input: ξ^{old}
Output: ξ^{new}

1: Calculate the approximate required capacity of each set $\zeta \in \xi^{\text{old}}$, and decide $\zeta^\star$ to be divided.
2: Calculate the approximate required capacity of each service $s \in \zeta^\star$, and decide $s^\star$ to be divided.
3: $q_{s^\star v^\star} \leftarrow q_{s^\star v^\star} + 1$.
4: **for** all $s^\star \in \zeta^*$, $\zeta^* \in \xi$ **do**
5: $\quad k_{s^\star}^{\zeta^*} \leftarrow \frac{1}{q_{s^\star v^\star}}$
6: **end for**
7: **if** there exists created set for VNF $v^\star$ that does not include $s^\star$ **then**
8: $\quad$ put the one new proportion of $s^\star$ to the created set.
9: **else**
10: $\quad$ put the one new proportion of $s^\star$ to an empty set.
11: **end if**
12: ξ^{old} is updated as ξ^{new}.
13: **if** any of the failure condition is met **then**
14: $\quad$ FPH-ES-d' fails.
15: **else**
16: $\quad$ Go to Step 1.
17: **end if**

At Step 3 of FPH-ES-d', the number of parts which the VNF belonging to each service is divided is set to 1, or $q_{sv} = 1, \forall s \in S, v \in V$, in the initial case. After the service to be divided is selected, the flows belonging to service $s^\star$ for VNF $v^\star$ are evenly divided into $q_{s^\star v^\star} + 1$ parts, where the one new part of flows is deployed on another VM and the proportions of all the old parts are updated. Step 3 repeats until the algorithm moves to Step 4, or meets the failure condition of the algorithm that the capacity of a whole VM is not enough to deploy one part of flows while all the services in the selected set reach the maximum admissible number of divisions. We consider both realistic and synthetic scenarios in Section V-D.

Figure 9 shows the service deployment cost and number of activated VMs obtained by ESH and FPH-ES of different traffic multiplier n in the realistic scenario. It is shown in Figs. 9(a) and 9(b) that FPH-ES outperforms ESH in all tested cases in terms of the service deployment cost and the number of activated VMs. This is because ESH mainly uses an optimization solver to solve the formulated MISOCP problem with limited computation time, which indicates that the obtained result is a feasible solution rather than the optimal one. Especially as n becomes large, the results of ESH are some feasible solutions obtained by the optimization solver which are far from the optimal one due to the limited computation time, which also explains the reason why the difference between ESH and FPH-ES increases as n is larger;

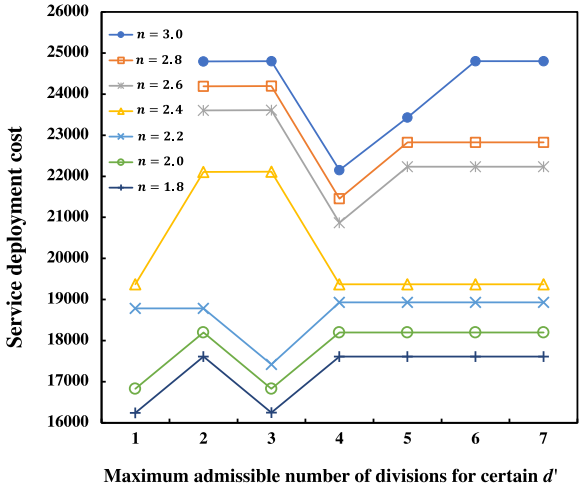


FIGURE 10. Dependency of service deployment cost on maximum admissible number of divisions for certain d' in FPH-ES-d'.

while FPH-ES, which is similar to FPH, is also a heuristic that decomposes the complicated problem into several simpler steps. As a result, it obtains solutions with relatively smaller cost within much shorter time in all tested cases.

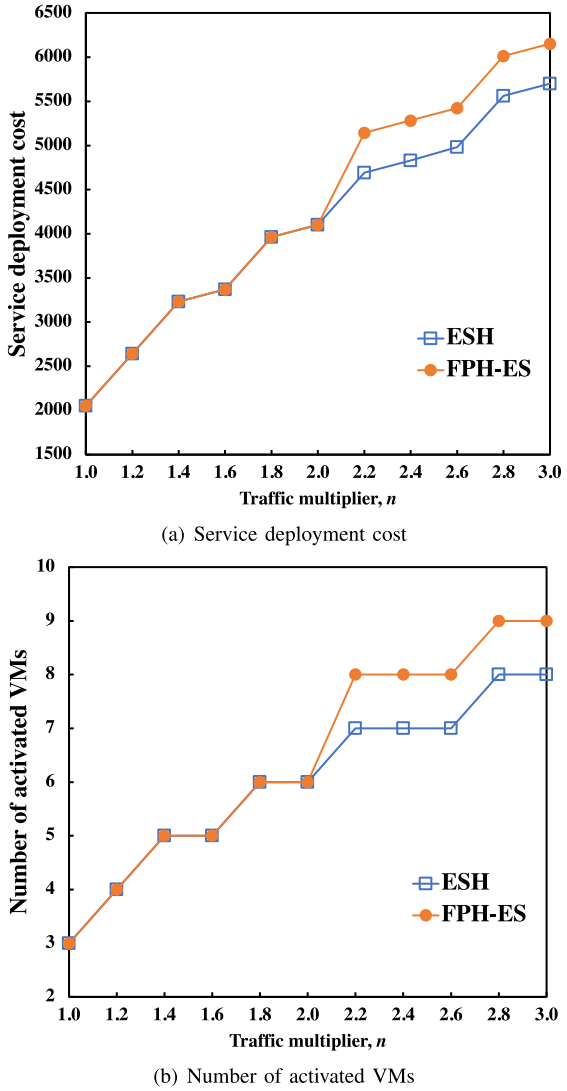
Table 6 presents the computation time to obtain the results of FPH-ES in Fig. 9. We observe that the computation time tends to become large as the traffic scale increases except for the case with $n = 2.2$. This is because the flows of VNFs are divided into more parts and deployed on more VMs as the traffic scale becomes large. We also notice that the computation time with $n = 2.2$ is much larger than those around it, which indicates that FPH-ES loops at Step 5 for a certain value of d'.

Figure 10 shows the service deployment cost obtained by FPH-ES-d' for a certain d' with different traffic multiplier n in the realistic scenario. We observe that the deployment cost reaches the lowest value at $d' = 3$ when $n = 1.8, 2.0,$ and 2.2, and it reaches the lowest value at $d' = 4$ when $n = 2.4, 2.6, 2.8,$ and 3.0. The reason why the deployment cost of both sides are higher is as follows: for the left side, it restricts the partition of flows due to the limited value of d', and therefore, the algorithm needs to deploy the whole part of flows on a new VM; for the right side, it allows one large flow to be divided into a larger number of parts, each of which requires one VM to deploy, and can leave other small flows not sufficiently divided. Both conditions lead to a fact that the algorithm tends to use the whole VM to deploy one part of flows, which causes the waste of VM capacity and the increase of deployment cost. We also notice that the value of d' with which the lowest deployment cost is obtained grows as n becomes large. This is because flow partition is applied in a larger degree with larger n, and the lowest point of d' in terms of the deployment cost, which can also be regarded as a tradeoff between the two conditions, also becomes large.

We should notice that the solution obtained in FPH-ES is only one possible case of ESH, which indicates that it does not perform better than the optimal solution of ESH.

TABLE 6. Computation time [s] for results of FPH-ES shown in Fig. 9.

n	1.0	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0
FPH-ES	0.93	1.10	1.49	2.50	2.52	3.91	11.66	3.93	4.66	4.84	4.97



(a) Service deployment cost

(b) Number of activated VMs

FIGURE 11. Comparison of service deployment cost and number of activated VMs between ESH and FPH-ES of different values of n in synthetic scenario.

Figure 11 shows the service deployment cost and number of activated VMs obtained by ESH and FPH-ES of different traffic multipliers n in the synthetic scenario, where optimal solutions of the MISOCP problem are obtained in ESH. It is shown in Figs. 11(a) and 11(b) that ESH outperforms FPH-ES when n ranges from 2.2 to 3.0, which indicates that ESH provides an assignment of equally divided proportions of flows that requires less deployment cost compared to the one obtained by FPH-ES in these cases. For other cases, FPH-ES obtains the solution with the lowest required deployment cost as ESH does so that the results are the same in ESH and FPH-ES. FPH-ES requires 4.0% more service deployment

TABLE 7. Computation time [s] for results shown in Fig. 11.

n	1.0	1.2	1.4	1.6	1.8	2.0
FPH-ES	0.06	0.08	0.11	0.09	0.09	0.08
ESH	119.36	330.45	597.84	534.45	1469.36	354.15

n	2.2	2.4	2.6	2.8	3.0
FPH-ES	0.09	0.09	0.13	0.10	0.10
ESH	20790.12	11129.84	41577.14	10048.97	13784.83

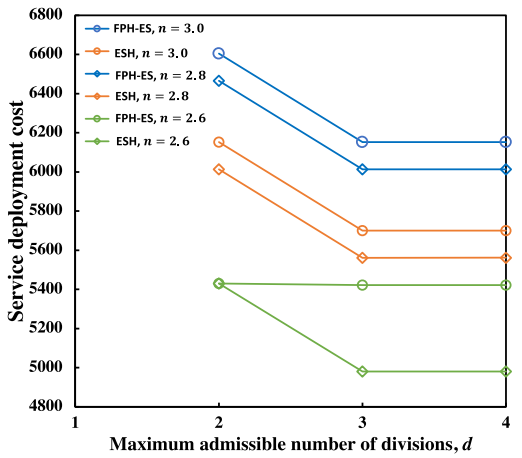


FIGURE 12. Comparison of service deployment cost between ESH and FPH-ES with dependency on maximum admissible number of divisions, d.

cost and 6.2% more activated VMs in average compared to ESH in the synthetic scenario.

Table 7 presents the computation time of results shown in Fig. 11. It shows that the computation time of FPH-ES is around 0.1 [s] in the synthetic scenario, which largely saves the computation time compared to ESH in all tested cases.

Figure 12 shows the comparison of service deployment cost between ESH and FPH-ES with the dependency on maximum admissible number of divisions, d, in the synthetic scenario. We observe that, under a given value of d, ESH obtains a solution with lower deployment cost compared to the one obtained by FPH-ES. It reveals that ESH outperforms FPH-ES under the same restriction on flexibility of flow partition in even-splitting cases.

VI. CONCLUSION

This paper proposed a service deployment model with flow partition to minimize the service deployment cost with satisfying the service delay constraint. We provided the mathematical formulation for the proposed model and transform its special case as an MISOCP problem to obtain the optimal solution. A flow partition heuristic based on problem decomposition, called FPH, was introduced to solve the problem in practical time. We compared the performances of proposed model with flow partition and conventional model without flow partition. We considered the formulated MISOCP problem with an even-splitting strategy in a special case as another heuristic of the original problem called ESH; the performances of FPH and ESH were compared in a realistic scenario. We also considered the formulated MISOCP problem as an original problem and compared it to an FPH-based heuristic algorithm with adopting the even-splitting strategy, called FPH-ES, in both realistic and synthetic scenarios. The numerical results showed that compared to the conventional model, the proposed model can save both initial and proportional cost; it improves the maximum admissible traffic scale by 23% in average in our examined cases. We evaluated the performances of FPH and FPH-ES with the dependency of maximum admissible number of divisions. We found that the service deployment cost is saved by increasing the maximum number of parts that flows can be divided to a certain point, but it does not keep decreasing after the point. We also observed that FPH outperforms ESH and ESH outperforms FPH-ES in terms of the service deployment cost in their own focused problems, respectively.

REFERENCES

[1] J. Zhang, F. He, and E. Oki, "Service deployment on shared virtual network functions with flow partition," in *Proc. IEEE ICC*, 2022, pp. 1–6.

[2] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, 1st Quart., 2016.

[3] F. He, T. Sato, and E. Oki, "Optimization model for backup resource allocation in middleboxes with importance," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1742–1755, Aug. 2019.

[4] A. Marotta, F. D'Andreagiovanni, A. Kassler, and E. Zola, "On the energy cost of robustness for green virtual network function placement in 5G virtualized infrastructures," *Comput. Netw.*, vol. 125, pp. 64–75, Oct. 2017.

[5] J. Pei, P. Hong, M. Pan, J. Liu, and J. Zhou, "Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 263–278, Feb. 2020.

[6] R. Gouareb, V. Friderikos, and A.-H. Aghvami, "Virtual network functions routing and placement for edge cloud latency minimization," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2346–2357, Oct. 2018.

[7] S. Agarwal, F. Malandrino, C. F. Chiasserini, and S. De, "VNF placement and resource allocation for the support of vertical services in 5G networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 433–446, Feb. 2019.

[8] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *Proc. IEEE CloudNet*, 2015, pp. 171–177.

[9] L. Qu, C. Assi, M. Khabbaz, and Y. Ye, "Reliability-aware service function chaining with function decomposition and multipath routing," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 2, pp. 835–848, Jun. 2020.

[10] J. Sun, F. Liu, M. Ahmed, and Y. Li, "Efficient virtual network function placement for poisson arrived traffic," in *Proc. IEEE ICC*, 2019, pp. 1–7.

[11] F. Malandrino, C. F. Chiasserini, G. Einziger, and G. Scalosub, "Reducing service deployment cost through VNF sharing," *IEEE/ACM Trans. Netw.*, vol. 27, no. 6, pp. 2363–2376, Dec. 2019.

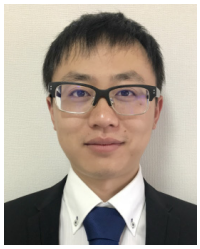
[12] F. He and E. Oki, "Robust virtual network function deployment against uncertain traffic arrival rates," in *Proc. IEEE NetSoft*, 2021, pp. 339–347.

[13] D. Michalopoulos, M. Doll, V. Sciancalepore, D. Bega, P. Schneider, and P. Rost, "Network slicing via function decomposition and flexible network design," in *Proc. IEEE PIMRC*, 2017, pp. 1–6.

[14] S. Rajagopalan, D. Williams, H. Jamjoom, and A. Warfield, "Split/Merge: System support for elastic execution in virtual Middleboxes," in *Proc. USENIX NSDI*, 2013, pp. 227–240.

[15] A. Pasic, P. Babarczi, J. Tapolcai, E. Berczi-Kovacs, Z. Kiraly, and L. Ronyai, "Minimum cost survivable routing algorithms for generalized diversity coding," *IEEE/ACM Trans. Netw.*, vol. 28, no. 1, pp. 289–300, Feb. 2020.
[16] A. Jerome, M. Yuksel, S. Ahmed, and M. Bassiouni, "SDN-based load balancing for multi-path TCP," in *Proc. IEEE INFOCOM WKSHPS*, 2018, pp. 859–864.
[17] M. R. Sama, X. An, Q. Wei, and S. Beker, "Reshaping the mobile core network via function decomposition and network slicing for the 5G era," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2016, pp. 1–7.
[18] D. Li, P. Hong, W. Wang, and J. Pei, "Virtual network function placement with function decomposition for virtual network slice," in *Proc. IEEE CSCN*, 2018, pp. 1–4.
[19] S. Prabhavat, H. Nishiyama, N. Ansari, and N. Kato, "On load distribution over multipath networks," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 3, pp. 662–680, 3rd Quart., 2012.
[20] T.-T.-L. Nguyen, T.-M. Pham, and H. T. T. Binh, "Adaptive multipath routing for network functions virtualization," in *Proc. SoICT*, 2016, pp. 222–228.
[21] Q. Wang, G. Shou, Y. Liu, Y. Hu, Z. Guo, and W. Chang, "Implementation of multipath network virtualization with SDN and NFV," *IEEE Access*, vol. 6, pp. 32460–32470, May 2018.
[22] Z. Xiang, F. Gabriel, E. Urbano, G. T. Nguyen, M. Reisslein, and F. H. P. Fitzek, "Reducing latency in virtual machines: Enabling tactile Internet for human-machine co-working," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1098–1116, May 2019.
[23] "IBM ILOG CPLEX optimization studio." IBM. Accessed: Nov. 15, 2022. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio
[24] "Gurobi optimizer 9.1." Gurobi. Accessed: Nov. 15, 2022. [Online]. Available: http://www.gurobi.com
[25] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, pp. 83–97, Mar. 1995.
[26] C. Casetti et al., "Arbitration among vertical services," in *Proc. IEEE PIMRC*, 2018, pp. 153–157.
[27] T. Taleb, A. Ksentini, and A. Kobbane, "Lightweight mobile core networks for machine type communications," *IEEE Access*, vol. 2, pp. 1128–1137, 2014.
[28] A. Ksentini, Y. Hadjadj-Aoul, and T. Taleb, "Cellular-based machine-to-machine: Overload control," *IEEE Netw.*, vol. 26, no. 6, pp. 54–60, Nov./Dec. 2012.

JINGXIONG ZHANG (Graduate Student Member, IEEE) received the B.E. degree from Shanghai University, Shanghai, China, in 2020, where he is currently pursuing the M.E. degree with Kyoto University, Kyoto, Japan. His research interests include optimization, resource allocation, and service deployment.



FUJUN HE (Member, IEEE) received the B.E. and M.E. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2014 and 2017, respectively, and the Ph.D. degree from Kyoto University, Kyoto, Japan, in 2020. His research interests broadly lie in communication and computer networks. Recently, his work focuses on network virtualization/softwarization, optical networks, and artificial intelligence for network control.

EIJI OKI (Fellow, IEEE) is a Professor with Kyoto University, Kyoto, Japan. He was with Nippon Telegraph and Telephone Corporation Laboratories, Tokyo, from 1993 to 2008, and The University of Electro-Communications, Tokyo, from 2008 to 2017. From 2000 to 2001, he was a Visiting Scholar with Polytechnic University, Brooklyn, NY, USA. His research interests include routing, switching, protocols, optimization, and traffic engineering in communication and information networks.