Service Deployment on Shared Virtual Network Functions with Flow Partition

Jingxiong Zhang, Fujun He, and Eiji Oki
Graduate School of Informatics, Kyoto University, Kyoto, Japan

***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 total deployment cost with meeting service time 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. A heuristic algorithm is introduced to solve the original problem in practical time by decomposing it into several steps; each step handles a convex problem. The numerical results reveal that the proposed model saves the total deployment cost compared to the conventional one. It improves the maximum admissible traffic scale by 23% in average in our examined cases.**

***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 [1], [2]. 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 [3]–[11]. 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 a 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 [12]. For a stateful virtual middlebox, such as an intrusion detection system, the work in [13] 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. Serval works utilized the SDN paradigm with multipath routing to achieve a high performance transmission, such as to improve the reliability with the diversity coding [14] or to perform load balancing [15].

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 total deployment cost with meeting the delay requirement of services. We set the maximum allowed number of partitions for each VNF in each service. We

This work was supported in part by JSPS KAKENHI, Japan, under Grant Number 21H03426.

provide the mathematical formulation for the proposed model. In order to solve the problem in practical time, a heuristic algorithm is introduced to decompose the problem into several steps; each step handles a convex problem. The numerical results show that compared to the conventional model, the proposed model saves the total deployment cost. It improves the maximum admissible traffic scale by 23% in average in our examined cases.

II. Related work

Several works considered VNF decomposition to improve the flexibility of resource utilization [5], [16]. In [16], the evolved packet core (EPC) user-plane function 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. The work in [5] introduced a service provisioning model with considering VNF decomposition, where two types of VNF decomposition were introduced. The VNF sharing among different services is not considered in [5]. In addition, the queueing and processing delays are assumed as a per-hop delay, which is given in [5]; 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 queuing systems to be aware of service delay more explicitly [10], [11]. The work in [10] adopted different approaches of traffic priority for VNF sharing. The model introduced in [10] considered the process of running a VNF on a VM as an M/M/1 system with priority queueing. The work in [11] solved the VNF deployment problem based on first-in-first-out queueing. The model presented in [11] took traffic uncertainty into consideration, where the traffic arrival rate is not always deterministic in practical applications.

Our work is 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 [17]. The work in [18] 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 [18] 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 [18], this work focuses on the VNF deployment and computing resource allocation with flow partition.

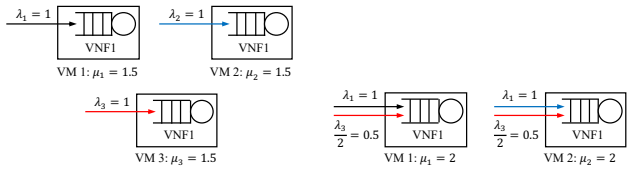
III. Model and problem formulation

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.

Fig. 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.

Fig. 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.



(a) Without flow partition (b) With flow partition

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

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 an nonempty 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: $T_s = \sum_{v \in V} t_{sv}$.

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. 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, which is expressed as:

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

All parts of flows are deployed on different VMs which host the VNFI for VNF v, which 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:

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

Equation (8) indicates that the total arrival rate of flows on VM m running VNF v 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:

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

3) Problem formulation: VNFs belonging to services need to be deployed such that all services are finished in the maximum admissible average delay. 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:

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

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

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

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

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

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

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

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^i_{sv}, z^m_{svi}, and w_{mv}, respectively.

IV. Heuristic algorithm

We introduce an algorithm as outlined in Fig. 2 to solve the problem in practical time. Fig. 3 shows an example to demonstrate different steps of the algorithm. 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 $\xi = \{\{s_1, s_2, s_3\}, \{s_2, s_3, s_4\}\}$.

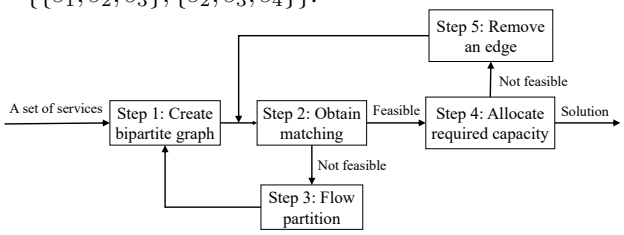


Fig. 2. Flowchart of the algorithm

A. Steps 1 and 2: Create bipartite graph and obtain minimum weight matching

At Step 1, the algorithm 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 that VM, including K_m^{f} and K_m^{u}, denotes the weight on the edge.

At Step 2, the algorithm finds an 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 [19] is used to solve the matching problem in polynomial time and obtains the matching whose total cost is minimized.

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 to divide one subset of ξ_v in ξ and activate new VMs to deploy the divided part. In order to meet the stability constraints of VMs, we approximately calculate the required capacity of each element in ξ and divide the one which requires the most. We assume that the maximum admissible time delay of service s is evenly distributed on each involved VNF; the admissible time delay of each VNF belonging to service s is expressed as: $\frac{d_s}{\sum_{v \in V} g_{sv}}$. Let k_s^{ζ} denote the proportion of VNF belonging to service s in set $\zeta \in \xi$. In the initial case, we set $k_s^{\zeta} = 1, \forall \zeta \in \xi, s \in \zeta$. Therefore, the maximum admissible time delay of element ζ is expressed as: $\min_{s \in \zeta} \frac{d_s}{\sum_{v \in V} g_{sv}}$, and the approximate required capacity is expressed as: $l_v(\frac{1}{\min_{s \in \zeta} \frac{d_s}{\sum_{v \in V} g_{sv}}} + \sum_{s \in \zeta} k_s^{\zeta} \lambda_{sv})$. 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 approximated 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 an empty set. 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: $A_{sv^{\star}} = l_{v^{\star}}(\frac{1}{\frac{d_s}{\sum_{v \in V} g_{sv}}} + k_s^{\zeta^{\star}} \lambda_{sv^{\star}})$. 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. Then we consider how to divide the VNF $v^{\star}$ belonging to $s^{\star}$. 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 $\Delta = \min_{s \in \zeta^{\star}} \frac{k_s^{\zeta^{\star}} \lambda_{sv^{\star}}}{d_{sv^{\star}}}$. If $U_{\zeta} = 0, \forall \zeta \in \xi$, and the algorithm does not obtain a feasible matching, the algorithm fails. After each time of division, the value of $k_{s^{\star}}^{\zeta^{\star}}$ is updated and $U_{\zeta^{\star}}$ is decreased by one. The algorithm goes back to Step 1. Considering improving usage efficiency of VMs, we define that in the new loop, if the selected element in ξ is the same as the one in previous loops, the algorithm 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, the algorithm enters Step 3. Let Δ_n denote the value of Δ in the nth loop. 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 $\Delta_1 = \frac{1}{3}\lambda_{s_3 v_1}$. 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 $k_{s_3}^{\zeta_{v_1}} = \frac{\lambda_{s_3 v_1} - \Delta_1}{\lambda_{s_3 v_1}} = \frac{2}{3}$, and $k_{s_3}^{\zeta_{v_1'}} = \frac{\Delta_1}{\lambda_{s_3 v_1}} = \frac{1}{3}$. Then the algorithm starts from Step 1. Suppose that the algorithm 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 $\Delta_2 = \frac{2}{9}\lambda_{s_3 v_1}$. Therefore, the divided part is moved from ζ_{v_1} to $\zeta_{v_1'}$, 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 $k_{s_3}^{\zeta_{v_1}} = \frac{\frac{2}{3}\lambda_{s_3 v} - \Delta_2}{\lambda_{s_3 v}} = \frac{4}{9}$, and $k_{s_3}^{\zeta_{v_1'}} = \frac{\frac{1}{3}\lambda_{s_3 v} + \Delta_2}{\lambda_{s_3 v}} = \frac{5}{9}$.

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, the algorithm 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 [20], [21].

If Step 4 fails, it indicates that the allocated capacity for VMs is not enough to satisfy the time 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

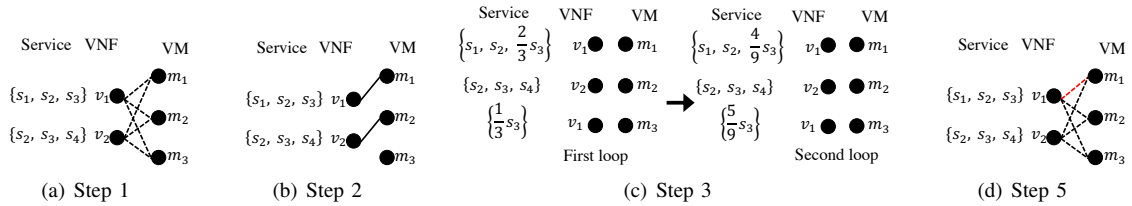


Fig. 3. Algorithm steps, where dashed lines in black and red represent possible and removed matchings, respectively; solid lines represent obtained matchings.

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; the algorithm restarts at Step 2 with a modified bipartite graph. As shown in Fig. 3(d), suppose that the algorithm obtains the minimum weight matching of the bipartite graph in Fig. 3(a), but the algorithm 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 the algorithm goes back to Step 2.

V. Numerical Results

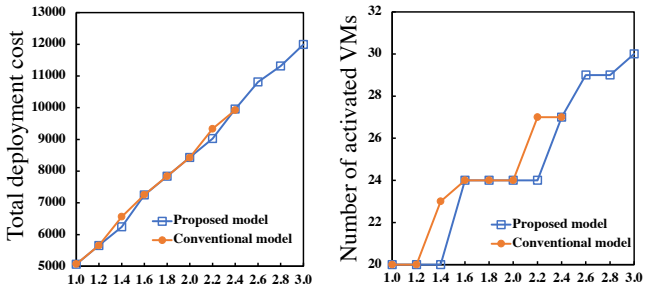
In this section, we compare the proposed model with the conventional model that does not consider flow partition. The two models are handled by the introduced heuristic and its modified version, respectively. In the modified version, we do not divide the flows of VNFI that belongs to the selected service at Step 3, but move the whole VNFI from the selected set to another set, or $k_s^\zeta = 1, \forall \zeta \in \xi, s \in S$. The SOCP problems of both models are solved by Gurobi with version 9.1.2. We use 1.8 GHz Dual-Core Intel Core i5 processor, 8GB memory in our evaluations.

A. Experiment settings

We consider a realistic scenario as introduced in [10]. In the realistic scenario, there are five services, where 17 VNFs are requested; the time delay of each service is set to 1. The arrival rates of flows and the distribution of VNFs that belong to each service are obtained from Table III in [10], where the coefficient of each VNF, or $l_v, v \in V$, is set to 1. We consider 30 VMs; the initial cost, the proportional cost, and the maximum capacity of each VM are 100, 1, and 450, respectively. We set a traffic multiplier, n, which ranges from 1.0 to 3.0 to regulate the scale of traffic rates.

B. Comparison between proposed and conventional models

Fig. 4 presents the total deployment cost and numbers of activated VMs in the proposed and conventional models of different values of multiplier n in the realistic scenario. Fig. 4(a) reveals that the total 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 4.9% and 3.3% of the deployment cost compared to the conventional model. In other cases, the deployment cost is comparable in two models. Fig. 4(b) shows the same tendency as Fig. 4(a), where the proposed model needs to activate 13% and 11% less VMs



(a) Total deployment cost (b) Number of activated VMs

Fig. 4. Comparison of total deployment cost and number of activated VMs between proposed and conventional models of different values of n. No feasible solution is obtained in the conventional model with $n \geq 2.6$.

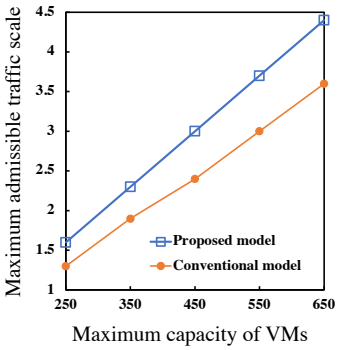


Fig. 5. Comparison of maximum admissible traffic scale between proposed and conventional models or different values of maximum VM capacity, C_m.

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

TABLE I
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	2.02	3.03	3.70	4.29	5.68	21.4	10.62	16.28	15.61	16.34	24.47
Conventional	0.24	0.41	0.59	0.62	0.62	0.62	1.31	1.40	—	—	—

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 total 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 the deployment. 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.

Table I presents the computation time to obtain the results in Fig. 4; and the hyphen symbol means that there is no feasible solution. It shows a tendency that the computation time increases as the value of n increases. This is because VNFIs of the same VNF are deployed on more VMs as the scale of flows gets larger in both models. We notice that the computation time of proposed model when n is 2.0 is higher than the value around it. This is because the algorithm enters the loop at Step 5 for more times. We also observe that the proposed model takes 11.55 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.

Fig. 5 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.

VI. CONCLUSION

This paper proposed the service deployment model with flow partition to minimize the total deployment cost with satisfying the service delay constraint. We provided the mathematical formulation for the proposed model. The heuristic algorithm based on problem decomposition was introduced to solve the problem in practical time. 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.

REFERENCES

[1] 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, 2016.

[2] 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, 2019.

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

[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, 2017.

[5] 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.*, 2019.

[6] 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, 2020.

[7] 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, 2018.

[8] 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.

[9] 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, 2019.

[10] 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, 2019.

[11] F. He and E. Oki, "Robust virtual network function deployment against uncertain traffic arrival rates," in *Proc. IEEE NetSoft*, 2021, pp. 1–9.

[12] 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.

[13] 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.

[14] 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, 2020.

[15] 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.

[16] 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.

[17] 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, 2012.

[18] 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. 32 460–32 470, 2018.

[19] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, pp. 83–97, March 1995.

[20] IBM. IBM ILOG CPLEX optimization studio. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio

[21] Gurobi. Gurobi optimizer 9.1. [Online]. Available: http://www.gurobi.com