Availability-Aware Service Provisioning with Backup Sub-chain-enabled Sharing

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

***Abstract*—This paper proposes an availability-aware service provisioning model with backup sub-chain-enabled sharing in network function virtualization to minimize the deployment cost. A sub-chain consists of a set of ordered VNFs that corresponds to a part of or the whole function chain of a service. Different from a conventional model in which a backup sub-chain is dedicated to protecting a primary sub-chain of one service, the proposed model allows the backup sub-chain sharing among services to reduce the deployment cost. Due to the complexity of the investigated problem, a heuristic is designed to tackle it. The numerical results show that the proposed model achieves lower deployment cost with satisfying the availability requirement than the conventional one.**

***Index Terms*—Network function virtualization, service function chaining, availability, backup resource sharing**

I. Introduction

Network function virtualization (NFV) technology provides an efficient and flexible way to implement and manage network functions (NFs) [1]. Traditionally, NFs, such as firewalls and load balancers are deployed on dedicated hardware. By leveraging NFV technology, NFs are decoupled from proprietary hardware and deployed on commodity servers, which are called virtual network functions (VNFs). A network service can be presented by a forwarding graph constructed by a set of ordered VNFs. Network operators can select the required VNFs to constitute VNF chains, which improves the efficiency and flexibility of service provisioning.

In NFV, services are usually provided in the form of service function chain (SFC). To provide services for customers, service providers need to deploy VNFs to constitute SFCs. Besides, the traffic of a service needs to traverse the required VNFs in a predefined order. Strict order constraints impose a VNF, say f_1, to be processed prior to another VNF, say f_2, that has dependency on f_1; some VNFs, for example, a firewall, may not have order constraints with other ones [2]. This feature provides flexibility to route through the required VNFs for service requests, which may reduce the number of deployed VNFs and helps to improve the resource usage.

Availability is a key performance indicator in measuring the service quality in the network [3]. While NFV brings advantages to resource utilization, it also introduces challenges to guarantee the service availability. On one hand, the availability of an SFC has end-to-end characteristics and needs to consider the availabilities of constituent VNFs and links [4]. On the other hand, the availability of a function in an SFC is related to the availabilities of both VNF and physical node hosting the VNF. Existing works for availability-aware SFC provisioning assume that either the link or the node in the substrate network (SN) never fails [5]–[8].

The work in [8] presented a service chain provisioning approach with sub-chain-enabled coordinated (SCEC) protection to satisfy availability requirements. The model in [8] divides a service into sub-chains; different types of protection can be applied to different sub-chains to meet the required availability in a cost efficient way. The model does not consider the failures of physical nodes, which may occur due to hardware problems including overloading [9]. Besides, it assumes the order constraints for all VNFs of service requests and applies the dedicated protection for each SFC. Recent studies [9]–[11] have shown that backup resource sharing can reduce the deployment cost to satisfy the availability requirement.

Based on the consideration in the previous paragraph, we address the availability-aware service provisioning with partial orders and backup sub-chain-enabled sharing in this paper. A question arises: how can we decide the processing order of VNFs for each service to facilitate the backup sub-chain sharing and decide the deployment of VNFs and the path through the VNFs in the SN to minimize the deployment cost with satisfying the required availability?

In this paper, we propose an availability-aware service provisioning model with backup sub-chain-enabled sharing in NFV. A sub-chain consists of a set of ordered VNFs that correspond to a part of or the whole function chain of a service. The proposed model allows the backup sub-chain sharing among services to reduce the deployment cost with satisfying the required availability. We compare the proposed model with a baseline that applies the SCEC model without backup sub-chain sharing. The numerical results show that the proposed model outperforms the baseline in terms of deployment cost.

The remainder of the paper is organized as follows. Section II describes the network model and motivation. Section III presents the proposed model with the method of availability calculation. A heuristic is designed in Section IV. Section V presents numerical results. Section VI summarizes the paper.

II. Network model and motivation

A. Network model

An SN consists of a set of nodes, V^{S}, and a set of directed links, L^{S}. Generally, the network nodes include facility

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

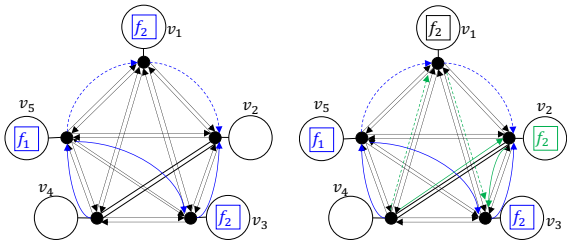
GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001366

nodes that hold computing capacity to provide functionality, networking nodes for communication between devices such as routers and switches, and other specialized hosts. This paper considers that each facility node is connected to the network via a router or switch. A facility node may fail due to several reasons such as hardware problems and overloading [12], which leads to the unavailability of services using VNFs deployed on this node. This paper focuses on the failures of facility nodes while the routers and switches are assumed not to fail [9]; the failure of a facility node causes the VNFs deployed on it to become unavailable while the traffic transmission can be performed through the connected router or switch. Let c_v and r_v^{node} denote the computing capacity and the availability of node $v \in V^{\text{S}}$, respectively. $(v_1, v_2) \in L^{\text{S}}$ denotes a directional link between $v_1 \in V^{\text{S}}$ and $v_2 \in V^{\text{S}} \setminus \{v_1\}$. Let directed graph $G^{\text{S}} = (V^{\text{S}}, L^{\text{S}})$ represent the SN. Let $b_{v_1v_2}$ and $r_{v_1v_2}^{\text{link}}$ denote the transmission capacity and the availability of link $(v_1, v_2) \in L^{\text{S}}$, respectively. Fig. 1(a) displays an SN, where a solid circle represents a switch or router, and a hollow circle next to the solid circle represents a facility node.

A set of services, S, needs to be provisioned in the given SN. Let T denote the set of types of VNFs that are available for service provisioning. Let a_t and r_t denote the required computing resource and the availability of deploying one instance of VNF $t \in T$, respectively. Service $s \in S$ is characterized by a source node, $v_s^{\text{start}} \in V^{\text{S}}$, a destination node, $v_s^{\text{end}} \in V^{\text{S}}$, and a set of VNFs, T_s, where $T_s \subseteq T$ and $|T_s| = n_s$. Let b_s^{req} and r_s^{req} denote the required transmission resource and availability of $s \in S$, respectively. For $t_1, t_2 \in T_s$, there is $t_1 \neq t_2$, which means that a type of VNF $t \in T$ appears at most once in T_s. The relative processing orders between VNFs in T_s are given.

B. Recalling SCEC protection

We recall the SCEC protection introduced in [8]. Fig. 1(a) shows a deployment example of service s_1 with SCEC protection in the given SN, where s_1 requests two VNFs, f_1 and f_2, in order. The given SN is a fully connected mesh network with five nodes, where every pair of nodes are connected by two directed links in opposite directions. v_4 and v_2 are designated as the source and destination nodes for the deployment of s_1, respectively. In Fig. 1(a), a pair of working and backup paths are provided from v_5, which hosts f_1, to destination node v_2, each of which is called a sub-chain. If the working sub-chain, which corresponds to blue solid lines and f_2 deployed on v_3, fails, the backup sub-chain, which corresponds to blue dashed lines and f_2 deployed on v_1, will be used for service provisioning. Different kinds of protection can be applied for increasing the service availability, which include application-layer protection, network-layer protection, and coordinated protection [13], [14]. The coordinated protection can provide replicas for VNFs and backup paths of passing through required VNFs. The coordinated protection that allows to switch the working sub-chain to the backup sub-chain is called a sub-chain-enabled coordinated, i.e., SCEC, protection.



(a) Deployment of s_1 (b) s_1 and s_2 share backup sub-chain

Fig. 1. Example of shared backup sub-chain.

C. Shared backup sub-chain and partial orders among VNFs

In Fig. 1(a), the backup sub-chain is dedicated for the protection of s_1. Next we illustrate the benefits of considering sharing backup sub-chain among services. For the sake of simplicity, we assume that each node in the SN has the same availability and computing capacity, where $r_v^{\text{node}} = 0.98$ and $c_v = 1$ for each node v, and each link has the same availability and transmission capacity, where $r_l^{\text{link}} = 0.99$ and $b_l = 5$ for each link l. The parameters for VNFs f_1 and f_2 are $r_{f_1} = 0.98$, $a_{f_1} = 1$, $r_{f_2} = 0.97$, and $a_{f_2} = 1$. The required availability and transmission capacity of s_1 are 0.94 and 1, respectively. The availability of s_1 in Fig. 1(a) is $R_{s_1} = r_l^{\text{link}} \times r_{f_1} \times r_v^{\text{node}} \times (1-(1-r_l^{\text{link}^2} \times r_{f_2} \times r_v^{\text{node}})^2) = 0.946$, which satisfies the requirement of s_1. A new service s_2, which only requests f_2, needs to be deployed in the SN; its required availability and transmission capacity are 0.99 and 1, respectively. v_4 and v_3 are designated as the source and destination nodes for the deployment of s_2, respectively.

We consider the deployment of s_2 in the SN. Due to the limited computing capacity of each node, v_2 and v_4 can be used to deploy f_2 for s_2. When f_2 is deployed on v_4 and link (v_4, v_3) is used for traffic transmission, the availability is $r_{f_2} \times r_v^{\text{node}} \times r_l^{\text{link}} = 0.941$, which cannot satisfy the requirement of s_2. When both v_2 and v_4 are used to deploy f_2 and links (v_4, v_2) and (v_2, v_3) are routing path for s_2, the availability is $(1-(1-r_{f_2} \times r_v^{\text{node}})^2) \times r_l^{\text{link}^2} = 0.978$, which still cannot satisfy the requirement of s_2. To deploy s_2 with required availability, we consider sharing backup resource of s_1 with s_2; the corresponding deployment is shown in Fig. 1(b), where f_2 deployed on v_1 is shared by s_1 and s_2. The green solid and dashed lines in Fig. 1(b) denote the working and backup paths of s_2, respectively; f_2 on v_2 is the VNF on the working path of s_2. Let $r_{\text{sub}}^{s_1\text{p}}$, $r_{\text{sub}}^{s_2\text{p}}$, $r_{\text{sub}}^{s_1\text{b}}$, and $r_{\text{sub}}^{s_2\text{b}}$ denote the availability of primary sub-chain of s_1, the availability of primary sub-chain of s_2, the availability of backup sub-chain of s_1, and the availability of backup sub-chain of s_2, respectively, all of which are equal to $r_l^{\text{link}^2} \times r_{f_2} \times r_v^{\text{node}}$. In Fig. 1(b), the availability of s_1 is $R_{s_1} = r_l^{\text{link}} \times r_{f_1} \times r_v^{\text{node}} \times (r_{\text{sub}}^{s_1\text{p}} + (1-r_{\text{sub}}^{s_1\text{p}}) \times r_{\text{sub}}^{s_1\text{b}} \times r_{\text{sub}}^{s_2\text{p}}) = 0.942$, and the availability of s_2 is $R_{s_2} = r_{\text{sub}}^{s_2\text{p}} + (1-r_{\text{sub}}^{s_2\text{p}}) \times r_{\text{sub}}^{s_2\text{b}} \times r_{\text{sub}}^{s_1\text{p}} = 0.991$. By sharing the backup resource, the availability of s_1 is still satisfied, and s_2 can be deployed in the SN with required availability.

The illustrative example reveals that backup resource sharing among services can reduce the deployment cost of services

with satisfied availability in a given SN. For a backup sub-chain with one or more VNFs, whether it can be shared by another service depends on both types and processing order of VNFs on it. Recent studies [2] have shown that partial processing order among VNFs helps to improve the resource usage as it may reduce the number of deployed VNFs in service provisioning. Arranging the processing order of VNFs properly for each service can facilitate the backup sub-chain sharing. Apparently, the decisions of processing order and backup sub-chain sharing for multiple services are not trivial and a model capable of making such decisions is required.

III. Proposed model

A. Model description

Given a set of services, S, since there exist several possibilities of passing through the required VNFs for each service, we need to determine the order of processing VNFs for each $s \in S$. Let λ_{st}^{i}, $s \in S, t \in T_s, i \in [1, n_s]$, denote a binary variable; λ_{st}^{i} is set to 1 if VNF t is the ith function to be processed for s and 0 otherwise. To achieve the target availability of accepted service s with SCEC protection, we need to divide the VNFs of s into n_s blocks including non-empty blocks and empty blocks. A non-empty block has at least one VNF, where an appropriate kind of protection and corresponding deployment for it need to be determined; an empty block contains no VNF and we do not need to consider its deployment. If the working sub-chain in a block is protected by the backup sub-chain, we call such a block a parallel block. Otherwise, only the working sub-chain is provided in a block, and we call such a block a serial block. Let ϕ_{sk} denote the kth block of s, where $k \in [1, n_s]$. Let x_{st}^{k} denote a binary variable, which is set to 1 if VNF $t \in T_s$ is in ϕ_{sk}, and 0 otherwise. Let u_{sk} denote a binary variable, which is set to 1 if block ϕ_{sk} is a parallel block, and 0 otherwise. For a parallel block, a backup sub-chain needs to be provided.

In light of the example and analysis in Section II, the proposed model allows parallel blocks of services to share the backup sub-chain. Let E denote the set of all possible shared backup sub-chains for S. Let ξ_{ϵ} denote a binary variable, which is set to 1 if shared backup sub-chain $\epsilon \in E$ protects more than one block, and 0 otherwise. Let τ_{sk}^{ϵ} denote a binary variable, which is set to 1 if the backup sub-chain of block ϕ_{sk} corresponds to shared backup sub-chain $\epsilon \in E$, and 0 otherwise. Let $\theta_{sks'k'}$ to denote a binary variable, which is set to 1 if blocks ϕ_{sk} and $\phi_{s'k'}$, $s, s' \in S, k \in [1, n_s], k' \in [1, n_{s'}]$, are parallel blocks and share the backup sub-chain, and 0 otherwise.

In addition to the above decisions that determine the VNF forwarding graph, we need to determine the deployment of VNFs and the routing path to pass through them in the SN. For service $s \in S$, let integer variable y_{st}^{v} denote the number of instances of VNF t deployed on v for the working sub-chain. For parallel block ϕ_{sk}, both working sub-chain and backup sub-chain need to be provided. Let $w_{sk}^{v_1 v_2}, (v_1, v_2) \in L^{\mathrm{S}}$, denote a binary variable, which is set to 1 if link (v_1, v_2) is on the routing path of working sub-chain of ϕ_{sk} and 0 otherwise.

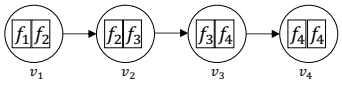


Fig. 2. Deployment example for availability calculation.

For the backup sub-chain in ϕ_{sk}, we use ${y^*}_{st}^{v}$ and ${w^*}_{sk}^{v_1 v_2}$ to describe its deployment, each of which has a meaning similar to the corresponding notation without *. Regarding the routing path for the deployment of a service, only one routing path is required except for the deployment of parallel blocks; let $W_s^{v_1 v_2}$ denote a binary variable, which is set to 1 if $(v_1, v_2) \in L^{\mathrm{S}}$ is used for the routing of s and does not belong to the routing path of any parallel block, and 0 otherwise.

To guarantee the availabilities of services sharing a backup sub-chain, the deployment of the shared backup sub-chain and corresponding primary sub-chains needs to be considered carefully. Firstly, the transmission capacity reserved on the routing path for the backup sub-chain needs to be sufficient for the recovery of an arbitrary protected primary sub-chain. Secondly, to avoid the correlated failures in the deployment of primary sub-chains, the routing paths of primary sub-chains need to be link disjoint and the VNF deployment of primary sub-chains cannot use the same substrate node.

The proposed model aims to minimize the deployment cost of services with satisfying the availability requirement. Different from the work in [8], where the deployment of services is independent from each other, the availabilities of services that share backup sub-chains are correlated.

B. Availability calculation

We illustrate the availability calculation method by the deployment of s_3 in Fig. 2, where s_3 requests f_1, f_2, f_3, and f_4 in order. Let V_{st} denote the set of deployed nodes for VNF $t \in T_s$ of service s; for s_3, there are $V_{s_3 f_1} = \{v_1\}$, $V_{s_3 f_2} = \{v_1, v_2\}$, $V_{s_3 f_3} = \{v_2, v_3\}$, and $V_{s_3 f_4} = \{v_3, v_4\}$. The availability of s_3 is $r_{s_3} = r^{\mathrm{path}} r^{\mathrm{VNFs}}$. Since the given deployment is serial, we have $r^{\mathrm{path}} = r_{v_1 v_2}^{\mathrm{link}} r_{v_2 v_3}^{\mathrm{link}} r_{v_3 v_4}^{\mathrm{link}}$. To ensure that s_3 is available, for each VNF of s_3, at least one node in the corresponding set of nodes on which the VNF is deployed survives and at least one replica of the VNF on the surviving nodes does not fail. Given the deployment, there are five possible combinations of surviving nodes, including $\{v_1, v_2, v_3\}$, $\{v_1, v_3\}$, $\{v_1, v_2, v_4\}$, $\{v_1, v_3, v_4\}$, and $\{v_1, v_2, v_3, v_4\}$. We have $r^{\mathrm{VNFs}} = r_{v_1}^{\mathrm{node}} r_{v_2}^{\mathrm{node}} r_{v_3}^{\mathrm{node}} (1 - r_{v_4}^{\mathrm{node}}) r_{f_1} (1 - (1 - r_{f_2})^2)(1 - (1 - r_{f_3})^2) r_{f_4} + r_{v_1}^{\mathrm{node}} (1 - r_{v_2}^{\mathrm{node}}) r_{v_3}^{\mathrm{node}} (1 - r_{v_4}^{\mathrm{node}}) r_{f_1} r_{f_2} r_{f_3} r_{f_4} + r_{v_1}^{\mathrm{node}} r_{v_2}^{\mathrm{node}} (1 - r_{v_3}^{\mathrm{node}}) r_{v_4}^{\mathrm{node}} r_{f_1} (1 - (1 - r_{f_2})^2) r_{f_3} (1 - (1 - r_{f_4})^2) + r_{v_1}^{\mathrm{node}} (1 - r_{v_2}^{\mathrm{node}}) r_{v_3}^{\mathrm{node}} r_{v_4}^{\mathrm{node}} r_{f_1} r_{f_2} r_{f_3} (1 - (1 - r_{f_4})^3) + r_{v_1}^{\mathrm{node}} r_{v_2}^{\mathrm{node}} r_{v_3}^{\mathrm{node}} r_{v_4}^{\mathrm{node}} r_{f_1} (1 - (1 - r_{f_2})^2)(1 - (1 - r_{f_3})^2)(1 - (1 - r_{f_4})^3)$, where each term corresponds to a possible combination of surviving nodes.

We describe the general method of availability calculation for services with given deployment. With SCEC protection, the VNFs of a service are divided into different blocks including serial blocks and parallel blocks. For block ϕ_{sk} of s, let r_{sk}^{w} denote the availability of the working sub-chain. The calculation of r_{sk}^{w} needs to consider the availabilities of both

VNFs in ϕ_{sk} and path connecting the VNFs on the working sub-chain; we have:

$$r_{sk}^{\mathrm{w}} = r_{sk}^{\mathrm{path}} r_{sk}^{\mathrm{VNFs}}. \tag{1}$$

The set of deployed nodes for VNFs on the working sub-chain in block ϕ_{sk} is $\boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{deploy}} = \cup_{t:x_{st}^k=1} V_{st}$. Let $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$ denote a subset of $\boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{deploy}}$ that enables the working sub-chain being available; in other words, $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$ is a set of nodes on which all types of VNFs in block ϕ_{sk} can be accessed. We define $\boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{sur}} = \{\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}\}$. We have:

$$r_{sk}^{\mathrm{VNFs}} = \sum_{\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}} \in \boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{sur}}} \Big(\prod_{v \in \boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}} r_v^{\mathrm{node}} \prod_{v \in \boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{deploy}} \setminus \boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}} (1 - r_v^{\mathrm{node}}) \prod_{t \in T_s : x_{st}^k = 1} \big(1 - (1 - r_t)^{\sum_{v \in \boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}} y_{st}^v}\big)\Big). \tag{2}$$

In (2), for each $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}} \in \boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{sur}}$, we obtain the product of the availability of the nodes in $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$ surviving, the availability of the nodes in $\boldsymbol{V}_{\boldsymbol{sk}}^{\mathbf{deploy}} \setminus \boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$ failing, and the availability of having at least one replica of each type of VNF in block ϕ_{sk} being available on the nodes in $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$; then we calculate the sum of the product corresponding to each $\boldsymbol{v}_{\boldsymbol{sk}}^{\mathbf{sur}}$.

In regarding the availability of a parallel block, it is necessary to consider the availabilities of both working and backup sub-chains. Let r_{sk}^{b} denote the availability of the backup sub-chain of block ϕ_{sk}. When ϕ_{sk} is a parallel block, we have:

$$r_{sk}^{\mathrm{path}} = \prod_{(v_1,v_2) \in L^{\mathrm{S}}} {r_{v_1 v_2}}^{w_{sk}^{v_1 v_2}}. \tag{3}$$

r_{sk}^{w} can be obtained by (1)-(3) and r_{sk}^{b} can be calculated in a similar way to r_{sk}^{w}. Let r_{sk}^{para} denote the availability of parallel block ϕ_{sk}. r_{sk}^{para} depends on whether the backup sub-chain is shared with other parallel blocks. We have:

$$r_{sk}^{\mathrm{para}} = r_{sk}^{\mathrm{w}} + (1 - r_{sk}^{\mathrm{w}}) r_{sk}^{\mathrm{b}} \prod_{s' \in S, k' \in [1, n_{s'}] : \theta_{sks'k'} = 1} r_{s'k'}^{\mathrm{w}}. \tag{4}$$

Except for the deployment of the parallel blocks, the rest of the deployment of a service is serial, including the deployment of serial blocks and the serial path connecting the adjacent blocks. Let r_s^{serial} denote the availability of the serial deployment of service s. We have:

$$r_s^{\mathrm{serial}} = \prod_{(v_1,v_2) \in L^{\mathrm{S}}} {r_{v_1 v_2}^{\mathrm{link}}}^{W_s^{v_1 v_2}} \prod_{k \in [1, n_s] : u_{sk} = 0} r_{sk}^{\mathrm{VNFs}}. \tag{5}$$

Let R_s denote the availability of s for the given deployment. R_s is given by:

$$R_s = r_s^{\mathrm{serial}} \prod_{k \in [1, n_s] : u_{sk} = 1} r_{sk}^{\mathrm{para}}. \tag{6}$$

C. Problem complexity

Theorem 1. *Given a set of services with relative processing orders of VNFs and an SN, the problem of availability-guaranteed service provisioning with the SCEC protection and backup sub-chain sharing to minimize the deployment cost is NP-hard.*

Algorithm 1 Simulated annealing (SA)

Input: G^{S}, S
1: Set $T = T^{\mathrm{init}}$
2: Randomly generate a possible set of values of $\boldsymbol{D}$ for services, where $\boldsymbol{D} = \{\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{\tau}\}$
3: Compute $C_{\boldsymbol{D}}$
4: **while** $T \geq T^{\mathrm{term}}$ **do**
5: Set $T = \sigma T$
6: Generate new set of $\boldsymbol{D'}$
7: Compute $C_{\boldsymbol{D'}}$
8: Set $\boldsymbol{D} = \boldsymbol{D'}$ and $C_{\boldsymbol{D}} = C_{\boldsymbol{D'}}$ with probability of $\min(1, e^{\frac{C_{\boldsymbol{D}} - C_{\boldsymbol{D'}}}{T}})$
9: **end while**
10: **return** $C_{\boldsymbol{D}}$

Proof: When the processing order of services is unique and the number of shared backup sub-chains among services is zero, the availability-guaranteed service provisioning with the SCEC protection has been proven to be NP-hard in [8]. Since the availability-guaranteed service provisioning without backup sub-chain sharing for given processing order of VNFs is a subset of the investigated problem, the availability-guaranteed service provisioning with the SCEC protection and backup sub-chain sharing is NP-hard. □

IV. Heuristic

Considering the complexity of the problem, we present a simulated annealing (SA) heuristic to solve it, which is given in Algorithm 1. SA is a metaheuristic to find near-optimal solutions for optimization problems [15]. In SA, running variable T denotes the "temperature" and parameter σ denotes the "cooling rate". In the beginning, T is set to T^{init}; an arbitrary feasible solution with a cost that corresponds to an objective function is given. In Algorithm 1, the solution is $\boldsymbol{D} = \{\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{\tau}\}$, which includes the processing order of each service, i.e., $\boldsymbol{\lambda} = \{\lambda_{st}^i : s \in S, t \in T_s, i \in [1, n_s]\}$, the block division of each service, i.e., $\boldsymbol{x} = \{x_{st}^k : s \in S, t \in T_s, k \in [1, n_s]\}$ and $\boldsymbol{u} = \{u_{sk} : s \in S, k \in [1, n_s]\}$, and the shared backup chains, i.e., $\boldsymbol{\xi} = \{\xi_\epsilon : \epsilon \in E\}$ and $\boldsymbol{\tau} = \{\tau_{sk}^\epsilon : \epsilon \in E, s \in S, k \in [1, n_s]\}$. The cost of $\boldsymbol{D}$ is $C_{\boldsymbol{D}}$. For given $\boldsymbol{D}$, $C_{\boldsymbol{D}}$ is obtained in Algorithm 2 by exploring the deployment cost of services with satisfied availability. In each iteration, T is set to σT; we obtain new solution $\boldsymbol{D'}$ by randomly changing the processing order of two VNFs for a random service based on existing solution $\boldsymbol{D}$. If $C_{\boldsymbol{D'}}$ is less than $C_{\boldsymbol{D}}$, we replace $\boldsymbol{D}$ by $\boldsymbol{D'}$; otherwise, $\boldsymbol{D'}$ is accepted to replace $\boldsymbol{D}$ with a probability of $e^{\frac{C_{\boldsymbol{D}} - C_{\boldsymbol{D'}}}{T}}$. Algorithm 1 is terminated when T is less than a given parameter T^{term}.

We explain Algorithm 2. Given solution $\boldsymbol{D}$, Algorithm 2 starts by initializing $C_{\boldsymbol{D}}$. Firstly, we deploy services one by one by using Algorithm 3. For each service, if the availability corresponding to the deployment satisfies its requirement, we increase the value of $C_{\boldsymbol{D}}$ with the deployment cost of the service. After all services have been deployed, we adjust the deployment of services based on $\boldsymbol{\xi}$ and $\boldsymbol{\tau}$, which denote the sharing of backup sub-chains. If two services share the

Algorithm 2 Cost calculation

Input: G^{S}, S, $\boldsymbol{D}$
$C_{\boldsymbol{D}} \leftarrow 0$
for $s \in S$ **do**
  Deploy s using Algorithm 3
  if $R_s \geq r_s^{\mathrm{req}}$ **then**
    Update $C_{\boldsymbol{D}}$
  end if
end for
for $s \in S$ **do**
  if s shares backup sub-chain with another service in S_1 **then**
    Adjust the deployment of s
    Recalculate R_s
    if $R_s \geq r_s^{\mathrm{req}}$ **then**
      Update $C_{\boldsymbol{D}}$ and the network status
    end if
  end if
end for
return $C_{\boldsymbol{D}}$

backup sub-chain, we need to adjust the deployments of their primary sub-chains to be link-disjoint. After the adjustment, we recalculate the availabilities of services that share the backup sub-chain. If the availability requirements of them are still satisfied, we update the deployment cost and network status. After all the adjustment, the value of $C_{\boldsymbol{D}}$ is returned.

We explain how to deploy a service in Algorithm 3. Given solution $\boldsymbol{D}$, Algorithm 3 deploys blocks of service $s \in S$ one by one. Let num_{st} denote the number of deployed instances of VNF $t \in T_s$, for s; initially num_{st} is set to 1 for VNFs in a serial block and is set to 2 for VNFs in a parallel block. Let v_{cur} denote the current node selected for deploying VNFs; v_{cur} is initialized as the source node of s. In each block, we deploy VNFs following the given processing order.

In Algorithm 3, if ϕ_{sk} is a serial block, we deploy VNF instances on v_{cur} as many as possible. Let c_v^{cur} denote the current available computing resource of node v. Let $d_{v_{\mathrm{cur}}v}$ denote the number of node hops on the path connecting v_{cur} and v with sufficient available transmission resource for the currently deployed service. We update v_{cur} by selecting node v from $V^{\mathrm{S}} \backslash \{v_{\mathrm{cur}}\}$ with the maximum value of $\frac{c_{v_{\mathrm{cur}}}^{\mathrm{cur}}}{d_{v_{\mathrm{cur}}v}}$ and record the used links for the route connecting the two nodes. Then we update the network status considering the used computing and transmission resources.

In Algorithm 3, if ϕ_{sk} is a parallel block, v_{cur} is selected as the ingress node of ϕ_{sk}. The deployment of a parallel block requires two link-disjoint paths between ingress and egress nodes of the parallel block with sufficient available transmission resource for s and sufficient computing resource for the VNF deployment of block ϕ_{sk}. For node $v' \in V^{\mathrm{S}} \setminus \{v_{\mathrm{cur}}\}$, we check whether there exists such link-disjoint paths between v_{cur} and v' with Suurballe's algorithm [16]; if there exists such paths, we add v' to set V' and record a pair of link-disjoint paths that has the least number of total node hops among all possible pairs of paths. After checking all nodes in $V^{\mathrm{S}} \setminus \{v_{\mathrm{cur}}\}$, we select $v' \in V'$ with the minimum total hops of the link-disjoint paths as the egress node of block ϕ_{sk}

Algorithm 3 Deployment of a service for given $\boldsymbol{D}$

Input: G^{S}, $\boldsymbol{D}$, s
Initialize R_s and $\mathrm{num}_{st}, \forall t \in T_s$.
while $R_s < r_s^{\mathrm{req}}$ and the maximum allowable number of iterations is not reached **do**
  $v_{\mathrm{cur}} = v_s^{\mathrm{start}}$
  for non-empty block ϕ_{sk} of s **do**
    if $u_{sk} = 0$ **then**
      while Any instance in ϕ_{sk} not deployed **do**
        Deploy instances on v_{cur}
        $v_{\mathrm{cur}} \leftarrow \arg\max_{v \in V^{\mathrm{S}} \setminus \{v_{\mathrm{cur}}\}} \frac{c_{v_{\mathrm{cur}}}^{\mathrm{cur}}}{d_{v_{\mathrm{cur}}v}}$
      end while
    end if
    if $u_{sk} = 1$ **then**
      Select the link disjoint paths
      Deploy instances of ϕ_{sk} on the selected paths
    end if
    if The deployment of ϕ_{sk} fails **then**
      Algorithm 3 is terminated
    else
      Update the resource usage in the network
    end if
  end for
  Connect v_{cur} and v_s^{end} with minimum hops
  if $R_s < r_s^{\mathrm{req}}$ **then**
    $\mathrm{num}_{st}++$ for VNF t with the smallest availability
  end if
end while

and update v_{cur} with v'. We deploy the VNF instances on the selected two paths between the ingress and egress nodes of block ϕ_{sk} based on Lemmas 1 and 3 in [13]. Then we update the resource usage in the network.

If Algorithm 3 cannot find any available node for deploying VNFs in a serial block or any available link-disjoint paths for deploying a parallel block, Algorithm 3 is terminated. After deploying the last non-empty block of s, we connect v_{cur} and the destination of s, i.e., v_s^{end}, through the path with the least number of node hops and sufficient transmission resource for s. We calculate the availability of s, i.e., R_s, based on the method presented in Section III-B. If R_s cannot satisfy the requirement of s, we select VNF t with the smallest availability among VNFs of s; the availability of VNF t is calculated by $1 - (1 - r_t)^{\mathrm{num}_{st}}$. The iterations terminate when R_s achieves the requirement or the maximum allowable number of iterations is reached.

V. Numerical results

We compare the proposed model with a baseline that applies the SCEC protection, which does not allow backup sub-chain sharing. We solve the two by Algorithm 1. For the baseline, we set $\xi_\epsilon = 0$ for each backup sub-chain $\epsilon \in E$ and set $\tau_{sk}^{\epsilon} = 0$ for each block ϕ_{sk} and ϵ. We implement the simulations on a computer equipped with an Intel Core i7-7700 3.60 GHz 4-core CPU, with 32G memory.

We apply the 24-node U.S. mesh network in Fig. 3, which has been used in [8] and [13], in the performance evaluation. We set the transmission capacity of each link and the computing capacity of each node to 500. The availabilities of

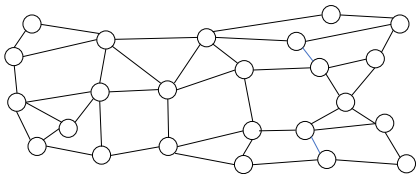


Fig. 3. 24-node U.S. mesh network.

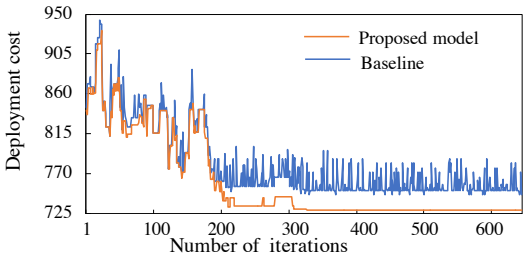


Fig. 4. Deployment costs of proposed model and baseline in one trial.

nodes and links are uniformly distributed over the range of [0.999, 0.99999]. There are 10 types of VNFs that are used to constitute services. The availabilities of VNFs are uniformly distributed over the range of [0.9, 0.999]. For VNF $t \in T$, the required computing resource, a_t, is an integer randomly chosen from the range of [1, 5] with equal probability. For each service, the number of VNFs is set to three and the availability requirement is 0.99. The source and destination nodes of a service are randomly selected from V^{S}, and the required transmission resource is an integer randomly chosen from the range of [1, 5] with equal probability.

We consider 15 services to be deployed with the aim of investigating the basic characteristics of proposed model and baseline. In each iteration of Algorithm 1, we record the deployment cost for given $\boldsymbol{D}$. For the proposed model and the baseline, all services are deployed successfully. Fig. 4 depicts the change in their deployment costs of the two in one trial. As the number of iteration increases, the deployment cost of each method decreases. With the same number of iterations, the proposed model achieves a lower deployment cost than the baseline due to the backup sub-chain sharing.

We consider different numbers of services, i.e., $|S|$, with adopting the average case analysis. For each $|S|$, we conduct 100 trials and obtain the average result from all trials. Table I reports the results, where MinBase and MinPro represent the minimum deployment cost obtained by the baseline and the proposed model, respectively, and BaseT and ProT represent the computation times of baseline and proposed time, respectively. Table I observes that the proposed model obtains a lower minimum deployment cost than the baseline for each given $|S|$, which verifies the effectiveness of the proposed model in reducing the deployment cost. The deployment cost of both methods increases as $|S|$ increases since more VNFs are deployed and more transmission resource are used for the route. BaseT and ProT increases with the increase of $|S|$ since the number of variables to be determined increases. For each $|S|$, ProT is larger than BaseT due to the exploration of backup sub-chain sharing.

TABLE I
COMPARISON FOR DIFFERENT NUMBERS OF SERVICES.

$\|S\|$	MinBase	MinPro	BaseT [sec.]	ProT [sec.]
5	225.39	223.82	31.87	38.59
10	457.63	453.00	102.66	106.16
15	680.42	671.05	152.97	168.19
20	920.08	907.86	192.27	195.16

VI. CONCLUSION

This paper proposed an availability-aware service provisioning model with backup sub-chain-enabled sharing in NFV to minimize the deployment cost. We designed a heuristic to solve it. We compared the proposed model with a baseline. The numerical results showed that the proposed model outperforms the baseline in terms of deployment cost.

REFERENCES

[1] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De 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, 2015.

[2] Z. Allybokus, N. Perrot, J. Leguay, L. Maggi, and E. Gourdin, "Virtual function placement for service chaining with partial orders and anti-affinity rules," *Networks*, vol. 71, no. 2, pp. 97–106, Sept. 2018.

[3] N. ISG, "Network functions virtualisation (NFV); reliability; report on models and features for end-to-end reliability," *ETSI GS NFV-REL*, vol. 1, p. v1, Apr. 2016.

[4] ——, "Network functions virtualisation (NFV); resiliency requirements," *ETSI Standard GS NFV-REL 001*, Jan. 2015.

[5] L. Qu, M. Khabbaz, and C. Assi, "Reliability-aware service chaining in carrier-grade softwarized networks," *IEEE J. Sel. Areas in Commun.*, vol. 36, no. 3, pp. 558–573, Mar. 2018.

[6] Y. Wang *et al.*, "Reliability-oriented and resource-efficient service function chain construction and backup," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 1, pp. 240–257, Mar. 2021.

[7] Y. Zhang, F. He, and E. Oki, "Availability-aware service chain provisioning with sub-chain-enabled coordinated protection," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2021, pp. 1–5.

[8] ——, "Service chain provisioning with sub-chain-enabled coordinated protection to satisfy availability requirements," *IEEE Trans. Netw. Service Manage.*, online available, 2021.

[9] F. He and E. Oki, "Backup allocation model with probabilistic protection for virtual networks against multiple facility node failures," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 3, pp. 2943–2959, Sep. 2021.

[10] R. Guerzoni, Z. Despotovic, R. Trivisonno, and I. Vaishnavi, "Modeling reliability requirements in coordinated node and link mapping," in *Proc. 33rd Int. Symp. Reliable Distrib. Syst.* IEEE, Oct. 2014, pp. 321–330.

[11] F. He, T. Sato, B. C. Chatterjee, T. Kurimoto, U. Shigeo, and E. Oki, "Robust optimization model for primary and backup resource allocation in cloud providers," *IEEE Trans. Cloud Comput.*, Jan. 2021.

[12] G. Aceto, A. Botta, P. Marchetta, V. Persico, and A. Pescapé, "A comprehensive survey on internet outages," *Journal of Network and Computer Applications*, vol. 113, pp. 36–63, Jul. 2018.

[13] J. Kong *et al.*, "Guaranteed-availability network function virtualization with network protection and VNF replication," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[14] I. M. Araújo, C. Natalino, H. Chen, M. De Andrade, D. L. Cardoso, and P. Monti, "Availability-guaranteed service function chain provisioning with optional shared backups," in *Proc. IEEE Int. Conf. on the Des. of Reliable Commun. Networks (DRCN)*, Mar. 2020, pp. 1–6.

[15] C. E. Leiserson, R. L. Rivest, T. H. Cormen, and C. Stein, *Introduction to Algorithms*. MIT press Cambridge, MA, 2009.

[16] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.