Availability-Aware Service Chain Provisioning with Sub-chain-enabled Coordinated Protection

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

***Abstract*—This paper proposes a sub-chain-enabled coordinated protection model for the availability-guaranteed service function chain (SFC) provisioning, which considers the availability of each component to constitute an SFC, including links and VNFs. Unlike conventional protection models providing certain protection for the whole chain, the proposed model configures sub-chains for each SFC and provides proper protection for each sub-chain to achieve the required availability cost-efficiently. We formulate the proposed model as an optimization problem to minimize the deployment cost. A heuristic is presented to tackle the problem. The numerical results show that the proposed model outperforms the conventional ones in terms of deployment cost.**

***Index Terms*—Network function virtualization, service function chaining, availability, protection**

I. Introduction

Network function virtualization (NFV) technology emerges as a promising way to implement and manage innovative services in an effective and dynamic way [1]. NFV decouples NFs from proprietary hardwares and enables software implementation of NFs, i.e., virtual network functions (VNFs), on commodity servers. By leveraging NFV, network operators are able to deploy network services in the form of service function chains (SFCs) by selecting a set of VNFs and steering traffic to pass them in the required order.

Availability is a key performance indicator in measuring the service quality of an SFC in the network [2]. In NFV environment, the availability of an SFC has end-to-end (E2E) characteristics [3], [4]. The E2E SFC behavior is a combination of behaviors of its constituent functional blocks, which includes individual VNFs and links connecting these VNFs. Therefore, the availability of an SFC needs to be estimated based on the availability of each constituent functional block and the topological connected patterns of the chain.

Both malfunction of a VNF and loss of connection between VNFs may result in the unavailability of an SFC; protection schemes can be applied to guarantee the availability of an SFC. VNFs that run in virtual machines or containers are prone to fail due to bugs, software crash, etc. Several backup instances of a VNF are provided to improve its availability in SFC provisioning [5]–[8]. However, the works in [5]–[8] ignore the availability of link that is used to transmit traffic among VNFs. Some works explore the availability-guaranteed SFC provisioning by considering both link and VNF availabilities [9], [10]. A backup path is adopted to improve the availability of connection between network components. Despite the availability improvement by providing redundancy for both VNF and network path, we observe that the schemes in [9], [10], which provide a backup path for the whole chain, may result in resource over provisioning.

This paper proposes a sub-chain-enabled coordinated protection model to guarantee the required availability in SFC provisioning, which considers both VNF and link availabilities. We formulate the proposed model as an optimization problem to minimize the deployment cost. We present a heuristic to solve the problem. We compare the proposed model by using the heuristic with two baselines, each of which applies a conventional protection model. The numerical results show that the proposed model achieves the lowest deployment cost among the three.

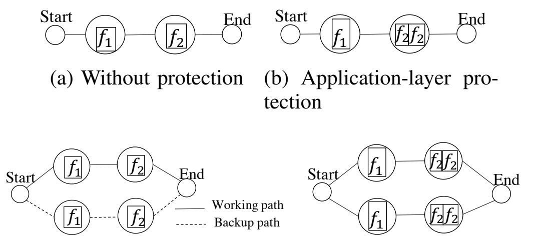
II. Motivation and problem statement

We investigate the availability-guaranteed SFC provisioning by considering the availabilities of both network-layer components (links) and application-layer components (VNFs). The physical server in the network is always reinforced or duplicated to ensure the normal operation. We make the simplified assumptions that instances of VNFs fail independently; for each specified VNF, its instances have the same availability and consume the same amount of computing resources [9].

We introduce three kinds of protection for increasing SFC availability [9], [10]. One choice is to provide several instances for a VNF, i.e., application-layer protection. Another choice is to provide network-layer protection, where a backup path passing through the required VNFs, i.e., backup service chain, is provided. Also, application-layer protection and network-layer protection can be applied in combination to improve the availability of an SFC, which is called coordinated protection.

We illustrate the three kinds of protection by s_1: $f_1 \to f_2$ in Fig. 1, where s_1 requests f_1 and f_2 in order. Fig. 1(a) shows a possible deployment of s_1 without any protection. In Fig. 1(b), two instances of f_2 are deployed for s_1; application-layer protection is applied. In Figs. 1(c) and 1(d), s_1 is protected with a backup service chain, where a dedicated path passes through the required VNFs. Different from Fig. 1(c), Fig. 1(d) adopts coordinated protection by providing both application-layer protection and network-layer protection. A conventional protection model either provides only working path similar to Fig. 1(b), or both working and backup paths for the whole service chain similar to Figs. 1(c) or 1(d).

We illustrate the availability difference among the above deployments in Fig. 1. Let r_l and r_f denote the availability of link l and VNF f, respectively. Let R_s denote the availability



(a) Without protection (b) Application-layer protection

(c) Network-layer protection (d) Coordinated protection

Fig. 1. Different kinds of protection for s_1.

requirement of SFC s. The availability requirement of s_1 is 0.96, i.e., $R_{s_1} = 0.96$. The parameters for the VNFs are $r_{f_1} = 0.98$ and $r_{f_2} = 0.97$. We assume that the availability of each link in the network is the same; it is $r_l = 0.99$ for link l.

Without any protection, the availability of s_1 in Fig. 1(a) is $R^a_{s_1} = {r_l}^3 \times r_{f_1} \times r_{f_2} = 0.99^3 \times 0.98 \times 0.97 = 0.922$. With the application-layer protection, the availability of s_1 in Fig. 1(b) is $R^b_{s_1} = {r_l}^3 \times r_{f_1} \times (1-(1-r_{f_2})^2) = 0.99^3 \times 0.98 \times (1-(1-0.97)^2) = 0.950$. With the network-layer protection, the availability of s_1 in Fig. 1(c) is $R^c_{s_1} = 1-(1-R^a_{s_1})^2 = 1-(1-{r_l}^3 \times r_{f_1} \times r_{f_2})^2 = 1-(1-0.99^3 \times 0.98 \times 0.97)^2 = 0.993$. With the coordinated protection in Fig. 1(d), the availability of s_1 is $R^d_{s_1} = 1-(1-R^b_{s_1})^2 = 1-(1-0.950)^2 = 0.997$.

Both deployments in Figs. 1(c) and (d) provide the required availability for s_1. Considering the cost, which includes the computing resource of VNF instances and transmission resource of links, the deployment in Fig. 1(c) has a lower cost than that in Fig. 1(d). The availability provided by Fig. 1(c) exceeds the availability required by s_1. Is it possible to further reduce the deployment cost and meet the availability requirement of s_1?

Next we introduce a more flexible protection, as shown in Fig. 2, than the protection in Fig. 1. In Fig. 2, a pair of working and backup paths are provided from f_1 to the end node, each of which we call a sub-chain; if the working sub-chain, which consists of f_2 and links, fails, the backup sub-chain will be used for service provisioning. The coordinated protection can be applied for sub-chains. We call the coordinated protection which allows to switch the working sub-chain to the backup sub-chain a sub-chain-enabled coordinated, i.e., SCEC, protection. The availability of s_1 in Fig. 2 is $R^{\text{SCEC}}_{s_1} = r_l \times r_{f_1} \times (1-(1-{r_l}^2 \times r_{f_2})^2) = 0.99 \times 0.98 \times (1-(1-0.99^2 \times 0.97)^2) = 0.967$. The SCEC protection in Fig. 2 satisfies the required availability of s_1 with a lower cost than those in Fig. 1, which shows the advantage of sophisticated protection on the sub-chains of an SFC.

When the SCEC protection is applied, a question arises. *How to configure sub-chains for an SFC and provide proper protection for each sub-chain to achieve the required availability of the SFC with minimum resource consumption?*

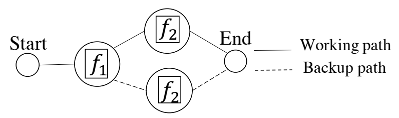


Fig. 2. Protection of sub-chain for s_1

III. Proposed Model

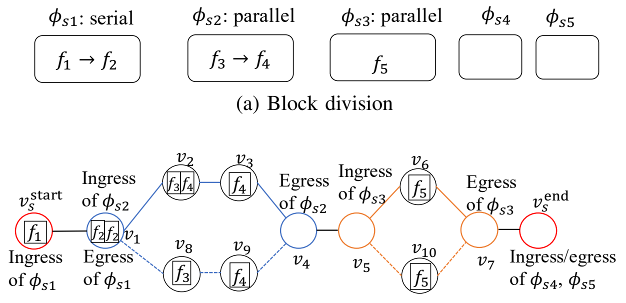
A. Network model

We are given a physical network represented by directed graph $G(V, L)$, where V is a set of nodes and L is a set of links connecting the nodes. Let c_v denote the limited computing resources of node $v \in V$. Let r_l and b_l denote the availability and transmission capacity of link $l \in L$, respectively. Link $l \in L$ can be represented by a pair of nodes (v_1, v_2), where $v_1 \in V$ and $v_2 \in V \setminus \{v_1\}$ are the tail and head of l, respectively.

A set of SFCs, S, needs to be provisioned in the given network. Let T denote the set of VNF types that are involved in the SFC provisioning. Let r_t and a_t denote the availability and the required computing resource of deploying one instance of VNF $t \in T$, respectively. Each $s \in S$ consists of n_s VNFs that must be processed in the specified order. Let (s, i) denote the ith function of $s \in S$, where $i \in [1, n_s]$. The set of VNF types of $s \in S$ is represented by $T_s = \{t^i_s : i \in [1, n_s]\}$. Let $v^{\text{start}}_s \in V$ and $v^{\text{end}}_s \in V$ denote the source and destination nodes of $s \in S$, respectively. The required availability and transmission capacity of $s \in S$ are r^{req}_s and b^{req}_s, respectively.

B. Description of proposed model

When the SCEC protection is applied for $s \in S$, the functions of s can be divided into n_s demarcated blocks, where a non-empty block consists of at least one function and an empty block contains no function; then we deploy VNFs for each non-empty block in the substrate network and connect the deployments of non-empty blocks in series to constitute the deployment of s. Let ϕ_{sk} denote the kth block of $s \in S$, and let T_{sk} denote the set of VNF types in ϕ_{sk}, where $k \in [1, n_s]$.



(a) Block division

(b) Physical deployment

Fig. 3. Example of SCEC protection for $s : f_1 \rightarrow f_2 \rightarrow f_3 \rightarrow f_4 \rightarrow f_5$.

There are two types of blocks according to the protection applied in a block: serial block and parallel block. If no protection or only application-layer protection is applied in $\phi_{sk} \in B$, a single path passes through all function types in T_{sk}

in the deployment; ϕ_{sk} is serial. If network-layer protection or coordinated protectction is applied in ϕ_{sk}, two paths, each of which passes through all function types in T_{sk}, are provided in the deployment; ϕ_{sk} is parallel.

We make four rules for block division. Firstly, if $\phi_{sk} \in B$ is not empty, any $\phi_{sk'} \in B$ with $k' < k$ is not empty. Secondly, if the index range of non-empty blocks of $s \in S$ is $[1, n_s^{\text{non-empty}}]$ with $n_s^{\text{non-empty}} < n_s$, ϕ_{sk} with $k \in [n_s^{\text{non-empty}}+1, n_s]$ is empty. Thirdly, for non-empty blocks ϕ_{sk} and $\phi_{sk'}$ of $s \in S$, where $k, k' \in [1, n_s], k \neq k'$, if there exists $t \in T_{sk}$ preceding $t' \in T_{sk'}$ on s, we have $k < k'$. Fourthly, at least one of adjacent non-empty blocks is parallel.

Fig. 3(a) shows a possible block division for $s : f_1 \rightarrow f_2 \rightarrow f_3 \rightarrow f_4 \rightarrow f_5$, which consists of three non-empty blocks and two empty blocks. For non-empty blocks, f_1 and f_2 belong to ϕ_{s1}, f_3 and f_4 belong to ϕ_{s2}, and f_5 belongs to ϕ_{s3}. ϕ_{s4} and ϕ_{s5} are empty blocks. Based on the rules of block division, the adjacent block of serial block ϕ_{s1}, i.e., ϕ_{s2}, is parallel, and parallel block ϕ_{s2} has a serial adjacent block, i.e., ϕ_{s1}, and a parallel adjacent block, i.e., ϕ_{s3}.

After the block division, we determine the deployment of each non-empty block; the deployment of working path for non-empty block ϕ_{sk} includes four aspects. Firstly, we select an ingress node and an egress node in the substrate network for each block to identify the scope of its deployment. Secondly, we determine the placement of (s, i) in ϕ_{sk}, which includes the number of instances and the nodes to host these instances. Thirdly, the route of working path is needed, which includes the route between instances of the same VNF and the route between adjacent VNFs in ϕ_{sk}. The deployment of backup path for a parallel block is similar to that of working path. Fourthly, if there is no VNF instance deployed on v_s^{start}, the route from v_s^{start} to the node where the first instance of the first VNF of s is needed; if there is no VNF instance deployed on v_s^{end}, the route from the node where the last instance of the last VNF of s to v_s^{end} is needed. Fig. 3(b) shows a possible physical deployment corresponding to the block division in Fig. 3(a) for s, where a circle represents a node in the network, and a rectangle in a node represents a VNF instance hosted by it. The source/destination nodes of s are marked in red.

We formulate the proposed model as an optimization problem to minimize the resource consumption. The objective is:

$$\min \sum_{v \in V} \sum_{s \in S} \sum_{i \in [1, n_s]} (y_{si}^v + {y^*}_{si}^v) a_{t_s^i} + \sum_{l \in L} \sum_{s \in S} [W_s^l + \sum_{k \in [1, n_s]} (w_{sk}^l + {w^*}_{sk}^l)] b_s^{\text{req}}, \tag{1}$$

where y_{si}^v and ${y^*}_{si}^v$ indicates the number of instances of (s, i) for the working path and backup path on $v \in V$, respectively; W_s^l indicates whether $l \in L$ is used in the serial part of the deployment; w_{sk}^l and ${w^*}_{sk}^l$ indicates whether $l \in L$ is used for the route of the working path and backup path in parallel block ϕ_{sk}, respectively. The constraints consist of the above the block division and the deployment of non-empty blocks, which are omitted due to the space limit.

Algorithm 1 Simulated annealing (SA)

```
Input: $V$, $a_v^t, \forall t \in T, v \in V$, $S$
 1: Set $T = T^{\text{init}}$
 2: Randomly generate a feasible set of $\boldsymbol{xu}$
 3: Compute $C_{\boldsymbol{xu}}$ (See Algorithm 2)
 4: while $T \geq T^{\text{term}}$ do
 5:     Set $T = \sigma T$
 6:     Generate new set of $\boldsymbol{x'u'}$
 7:     Compute $C_{\boldsymbol{x'u'}}$
 8:     Set $\boldsymbol{xu} = \boldsymbol{x'u'}$ and $C_{\boldsymbol{xu}} = C_{\boldsymbol{x'u'}}$ with probability of
        $\min(1, e^{\frac{C_{\boldsymbol{xu}} - C_{\boldsymbol{x'u'}}}{T}})$
 9: end while
10: return $C_{\boldsymbol{xu}}$
```

Algorithm 2 Cost calculation

```
Input: $G(V, L)$, $S$, $\boldsymbol{xu}$
 1: num$_{si}$=1, $\forall (s, i) \in A$
 2: while $r_s \geq r_s^{\text{req}}, \forall s \in S$ is not satisfied do
 3:     Solve the optimization problem with the objective (1)
 4:     for $s \in S$ do
 5:         Calculate $r_s = r_s^{\text{serial}} r_s^{\text{parallel}}$
 6:         if $r_s < r_s^{\text{req}}$ then
 7:             num$_{si}$ + + for $(s, i)$ with the weakest availability
 8:         end if
 9:     end for
10: end while
11: return $C_{\boldsymbol{xu}}$
```

C. Heuristic

We introduce a simulated annealing (SA) heuristic to solve the problem. SA is a metaheuristic to find near-optimal solutions for optimization problems [11]. In SA, T is a running variable indicating the "temperature", which is initialized to T^{init} at the beginning; σ is a parameter indicating the "cooling rate". The algorithm begins with an arbitrary feasible solution with a cost computed with respect to an objective function. In Algorithm 1, the solution is the block division for each SFC, i.e., $\boldsymbol{xu} = \{x_{si}^k, u_{sk} : (s, i) \in A, \phi_{sk} \in B\}$, where x_{si}^k is set to 1 if (s, i) is in ϕ_{sk}, and 0 otherwise; u_{sk} is set to 1 if ϕ_{sk} is parallel, and 0 otherwise. The cost of $\boldsymbol{xu}$ is $C_{\boldsymbol{xu}}$. For given $\boldsymbol{xu}$, $C_{\boldsymbol{xu}}$ is obtained in Algorithm 2 by exploring the minimum deployment cost to achieve the required availability of each SFC. In each iteration, T is set to σT; new solution $\boldsymbol{x'u'}$ is obtained by randomly changing the block division for a random SFC based on the existing solution. $\boldsymbol{x'u'}$ is accepted to replace $\boldsymbol{xu}$, if $C_{\boldsymbol{x'u'}}$ is less than $C_{\boldsymbol{xu}}$; otherwise, it is accepted with a probability, which depends on the difference between $C_{\boldsymbol{x'u'}}$ and $C_{\boldsymbol{xu}}$ and the value of T. The algorithm is terminated when T is less than a given parameter T^{term}.

We explain Algorithm 2. Given $\boldsymbol{xu}$, Algorithm 2 incrementally exploits VNF redundancy to satisfy the availability requirements for all SFCs [7] and calculates the corresponding deployment cost. The availability of $s \in S$ depends on the number of redundant instances of each VNF on s and the links used for the routing among these VNF instances. We define $\text{num}_{si} = \sum_{v \in V} (y_{si}^v + {y^*}_{si}^v)$ to denote the minimum number of instances of $(s, i) \in A$. Algorithm 2 starts by initializing $\text{num}_{si} = 1$ for each $(s, i) \in A$ and then enters the iterations. In each iteration, we solve the optimization problem to obtain a deployment for S. For each $s \in S$, we calculate its availability, i.e., r_s, based on the current deployment; if r_s does not meet the availability requirement, we select (s,i) with the weakest availability among VNFs of s, which means that (s,i) has the smallest value of $1-(1-r_{t_s^i})^{\sum_{v\in V}(y^v_{si}+y^{*v}_{si})}$ among VNFs of s, and increase num_{si} by one. The iterations terminate when the availability requirements of all SFCs are satisfied.

IV. Numerical results

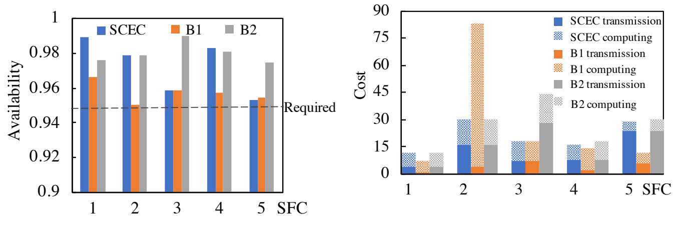
We compare the proposed model with two baselines, each of which uses a corresponding conventional protection model similar to [9], [10]. Baseline 1 provides only working path for each SFC, which applies no protection or the application-layer protection. Baseline 2 provides both working and backup paths for the whole chain of each SFC, which applies the network-layer protection or the coordinated protection. For baseline 1, we set $x^1_{si} = 1$ for each $(s,i) \in A$ and set $u_{s1} = 0$ for each $s \in S$. For baseline 2, we set $x^1_{si} = 1$ for each $(s,i) \in A$ and set $u_{s1} = 1$ for each $s \in S$. We obtain the deployment and corresponding cost for baselines 1 and 2 by Algorithm 2. The optimization problem is solved by IBM(R) ILOG(R) CPLEX(R) Interactive Optimizer 12.7.1.0 [12]. All the simulations are implemented on a computer equipped with an Intel Core i7-7700 3.60 GHz 4-core CPU, with 32G memory.

We consider a fully meshed network with five nodes to compare the basic characteristics of proposed model and two baselines. The computing capacity of each node and the transmission capacity of each link are integers randomly chosen from the range of [50,100] with equal probability. The availability of each link is set to a value randomly chosen from $C = \{0.95, 0.96, 0.97, 0.98, 0.99\}$ with equal probability. We consider five SFCs to be deployed. For each $s \in S$, the required availability and the number of VNFs are set to 0.95 and three, respectively; v_s^{start} and v_s^{end} are randomly chosen from V, and the required transmission resource is an integer randomly chosen from the range of [1,10] with equal probability. For one instance of VNF $t \in T$, the computing resource consumption is an integer randomly chosen from the range of [1, 5] with equal probability, and the availability is randomly chosen from C with equal probability.

We report the costs for SCEC and the two baselines in Table I. Considering that Algorithm 2 needs to solve the MILP problem, we set the maximum allowable computation time to one hour. Table I observes that SCEC outperforms the two baselines in terms of the total cost. Specifically, SCEC has a more balanced cost allocation of transmission resource and computing resource compared to the two baselines. For baseline 1, the only way to improve availability is to increase the number of instances of VNFs, so the computing resource consumption accounts for most of the total cost. Compared to baseline 1, the network-layer protection in baseline 2 reduces the computing resource cost, and increases the transmission resource cost. SCEC enables proper protection for different sub-chains, which optimizes the resource utilization to achive the required availabilities for SFCs with the lowest cost.

TABLE I
Comparison of deployment cost.

Model	SCEC	Baseline 1	Baseline 2
Total cost	105	134	134
Computing resource cost	46	114	54
Transmission resource cost	59	20	80



(a) Availability (b) Cost

Fig. 4. Availability and cost of each SFC.

Fig. 4 depicts the achieved availability relative to the required availability and the cost of each SFC in the obtained solutions of SCEC and the two baselines, where the results for baselines 1 and 2 are identified by B1 and B2, respectively. From Fig. 4, we observe that SCEC and baseline 2 have the same deployment cost for SFC 1; nevertheless, SCEC achieves higher availability of SFC 1 than baseline 2. Hence, with the same cost for an SFC, the availabilities of applying different kinds of protection can be different. Besides, more deployment cost does not necessarily lead to higher availability. Baseline 1 has the highest cost for SFC 2 but obtains the lowest availability of SFC 2 among the three; comparing the transmission resource and computing resource costs of SFC 2 in Fig. 4(b), it is clear that the network-layer protection for SFC 2 can achieve more availability improvements at lower cost than the application-layer protection. Consequently, highly sophisticated protection on the sub-chains in the SCEC model obtains the required availability at the lowest cost.

V. Conclusion

This paper proposed a sub-chain-enabled coordinated protection model for the availability-guaranteed SFC provisioning, which considers both link and VNF availabilities. We formulated the proposed model as an optimization problem to minimize the deployment cost. We presented a heuristic to tackle the problem. We compared the proposed model with two baselines. The numerical results showed that the proposed model has the lowest deployment cost among the three. We leave improving the efficiency of solving the proposed model in terms of computation time as our future work.

Acknowledgment

This work was supported in part by JSPS KAKENHI, Japan, under Grant 18H03230.

References

[1] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Commun. Mag.*, vol. 53, no. 2, pp. 90–97, 2015.

[2] N. ISG, "Network functions virtualisation (NFV); reliability; report on models and features for end-to-end reliability," *ETSI Standard GS NFV-REL 003*, 2016.

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

[4] S. Bijwe *et al.*, "End-to-end reliability assurance of service chain embedding for network function virtualization," in *2017 IEEE Conf. on Netw. Function Virtualization and Softw. Defined Networks (NFV-SDN)*. IEEE, 2017, pp. 1–4.

[5] J. Fan, M. Jiang, and C. Qiao, "Carrier-grade availability-aware mapping of service function chains with on-site backups," in *2017 IEEE/ACM 25th Int. Symp. on Qual. of Service (IWQoS)*. IEEE, 2017, pp. 1–10.

[6] J. Fan *et al.*, "Availability-aware mapping of service function chains," in *IEEE INFOCOM 2017-IEEE Conf. on Comput. Commun.* IEEE, 2017, pp. 1–9.

[7] L. Qu *et al.*, "A reliability-aware network service chain provisioning with delay guarantees in NFV-enabled enterprise datacenter networks," *IEEE Trans. on Netw. and Service Manage.*, vol. 14, no. 3, pp. 554–568, 2017.

[8] M. Wang *et al.*, "Availability-aware service chain composition and mapping in NFV-enabled networks," in *2019 IEEE Int. Conf. on Web Services (ICWS)*. IEEE, 2019, pp. 107–115.

[9] J. Kong *et al.*, "Guaranteed-availability network function virtualization with network protection and VNF replication," in *GLOBECOM 2017-2017 IEEE Global Commun. Conf.* IEEE, 2017, pp. 1–6.

[10] I. M. Araújo *et al.*, "Availability-guaranteed service function chain provisioning with optional shared backups," in *2020 16th Int. Conf. on the Des. of Reliable Commun. Networks DRCN 2020*. IEEE, 2020, pp. 1–6.

[11] C. E. Leiserson, R. L. Rivest, T. H. Cormen, and C. Stein, *Introduction to algorithms*. MIT press Cambridge, MA, 2001, vol. 6.

[12] IBM ILOG CPLEX optimization studio. https://www.ibm.com/products/ilog-cplex-optimization-studio. Accessed: Dec. 4, 2019.