Availability-Guaranteed Service Function Chain Provisioning with Optional Shared Backups

Availability-Guaranteed Service Function Chain Provisioning with Optional Shared Backups

Igor M. Araújo*†,Carlos Natalino†, Hao Chen‡, Marilet De Andrade‡, Diego L. Cardoso*, and Paolo Monti†
*Laboratory of Computational Intelligence, Federal University of Pará (UFPA), Belém, PA, Brazil
E-mail: {igoraraujo, diego}@ufpa.br
†Department of Electrical Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden
E-mail: {meireles, carlos.natalino, mpaolo}@chalmers.se
‡KTH Royal Institute of Technology, SE-164 40 Kista, Sweden
E-mail: {haoch, marilet}@kth.se

***Abstract*—The dynamic provisioning of Service Function Chain (SFC) using Virtual Network Functions (VNFs) is a challenging problem, especially for availability-constrained services. The provisioning of backup resources is often used to ensure that availability requirements are fulfilled. However, the assignment of backup resources should be carefully designed to avoid resource inefficiencies as much as possible.**

This paper proposes the Optional Backup with Shared Path and Shared Function (OBSPSF) strategy, which aims at improving resource efficiency while fulfilling the availability requirements of SFC requests. The strategy uses optional backup provisioning to ensure that backup resources are assigned only when strictly needed (i.e., when the SFC alone does not meet the availability constraint). Moreover, OBSPSF encourages backup sharing (among both connectivity and backup VNFs) to reduce the backup resource overhead. Results show that the strategy can accommodate orders-of-magnitude more services than benchmark heuristics from the literature.

***Index Terms*—Service Function Chaining, Virtualized Network Function, Provisioning, Availability, Shared Protection.**

I. Introduction

Traditional infrastructure providers deploy telecom services (e.g., Web Service, VoIP, and Video Conference [1]) using dedicated middle-boxes, i.e., equipment designed specifically to run a given set of network functions [2]. Being the hardware conceived with a specific purpose in mind, this approach has the advantage of guaranteeing the desired level of performance. On the other hand, the cost of using dedicated hardware does not scale well in 5th Generation of Networks (5G)-like scenarios where several services (i.e., verticals, each one with their specific requirements) should be accommodated in an agile way over the same network infrastructure over time. More specifically, infrastructure providers might have to over-provision middle-boxes in order to account for the worst-case scenario. This approach translates in possibly having, at any point in time, hardware resources seating idle and underutilized.

The work of I. M. Araújo is supported by the National Council for Scientific and Technological Development (CNPq Brazil) and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. This work was supported by VINNOVA as part of the project "Smart city concepts in Curitiba - Low-carbon transport and mobility in a digital society". M. De Andrade is now with Ericsson Research, Kista, Sweden.

One solution to the above problem is to deploy Commercial-Off-The-Shelf (COTS) hardware over which Virtual Network Functions (VNFs) can be run following the Network Function Virtualization (NFV) concept [3]. Traditional telecom services can then be re-defined in terms of SFCs where virtualized network functions are dynamically composed and provisioned to accommodate the requirements (e.g., connectivity, processing, availability) of a given service. This allows a more cost efficient use the connectivity and compute resources in the network infrastructure.

When looking at the requirements of most of the 5G verticals, availability is among the most critical and it needs to be carefully addressed. For this reason, the provisioning of SFCs with availability guarantees has attracted a lot of attention from the research community. Some works explored strategies for the on-site-only provisioning of backup VNFs [4] (i.e., at the same DataCenter (DC) location where the primary VNFs are running), an approach which improves service availability but does not protect against connectivity and/or DC site failures. If on-site-only VNF backup is not sufficient to achieve the required service availability level, additional and dedicated backup resources can be deployed to replicate the entire SFC. This means adding extra connectivity and backup VNFs resources, where the latter are deployed off-site (i.e., at a different DC location from the one where the primary VNFs are running) [5], [6]. On the other hand, it is well known from the literature that relying on dedicated backup resources is not resource efficient, while shared backup strategies can improve the resource utilization. Following this rationale, the authors in [7], [8] propose a backup strategy where the VNFs of the same SFC can be shared in the same off-site location, while no protection against the failure of connectivity resources is provided. None of the works mentioned above considered the possibility of sharing, among different SFCs, both backup connectivity and (off-site) VNFs resources so that services are protected against connectivity and DC failures while mitigating the resource overhead that come by adding backup resources.

This work proposes the Optional Backup with Shared Path and Shared Function (OBSPSF) strategy, a resource efficient approach that dynamically provisions availability-guaranteed

SFCs. The strategy introduces a cost function that for each candidate SFC provisioning solution *(i)* assigns backup resources only when strictly needed (i.e., when the primary SFC alone does not meet the availability constraint), and *(ii)* reduces the resource consumption of the backup SFCs by encouraging resource sharing of both the connectivity and the (off-site) VNFs backup resources. Results show that Optional Backup with Shared Path and Shared Function (OBSPSF) can drastically reduce the blocking probability in low-to-medium loads compared to traditional benchmark heuristics taken from the literature.

II. Problem Statement

This paper investigates the provisioning of availability-guaranteed SFC with efficient backup resource assignment. The inputs to the problem are: *(a)* the optical network topology comprising a set of optical nodes (i.e., optical cross-connects - OXCs) and fiber links interconnecting the nodes; *(b)* the availability associated with each link and node in the topology; *(c)* the set of DCs hosted by a subset of network nodes; and *(d)* the SFC to be provisioned, comprised of source and destination nodes, data rate, end-to-end availability requirement and VNFs to be deployed along the SFC path. The network capacity (i.e., available on fiber links and required by SFCs) is defined in terms of *connectivity units*. In the specific case of considering Wavelength Division Multiplexing (WDM) optical networks, a connectivity unit is equivalent to a wavelength. The processing capacity of DCs is defined in terms of *processing units*.

Given these inputs, the provisioning of an SFC consists of finding a path from the source to the destination with enough connectivity resources, and that traverses at least one DC with enough available processing resources to host the VNFs composing the chain. Moreover, this path should fulfill the availability requirements of the SFC. If a single chain does not meet the SFC required availability, the provisioning strategy can use protection to improve availability. The protection is realized by finding a pair of node- and link-disjoint paths (primary and backup). In this case, both primary and backup paths should each fulfill the connectivity and processing capacity required by the SFC, and the availability resulting from the path pair should fulfill the availability requirements of the SFC.

When backup resources are provisioned, they remain idle until some of the components in the primary SFC fail. However, reserving backup resources dedicated to a single SFC results in poor resource efficiency, leading to several side effects such as high blocking probability. Therefore, backup resource sharing is an appropriate strategy to mitigate the effects of reserving backup resources while guaranteeing the fulfillment of availability requirements.

Fig. 1 shows an example of the provisioning of 2 SFCs inspired by the algorithm proposed in [5]. The SFC request *SFC1* has the source at node 9 and destination at node 6. The strategy selects primary resources used for connectivity as links 9-7 and 7-6, and the DC co-located with node 7 for processing the VNFs. Let us assume that the availability offered by the primary SFC does not meet the required

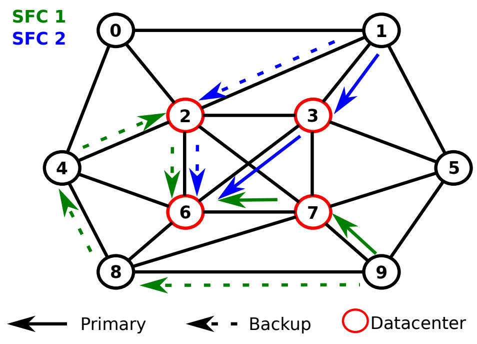


Fig. 1: SFC provisioning example

availability. One way to increase the availability of *SFC1* is by provisioning a backup SFC. In this example, the backup SFC of *SFC1* is composed of links 9-8, 8-4, 4-2, and 2-6, with node 2 hosting the VNFs. Then, a second SFC request *SFC2* arrives, with source at node 1 and destination at node 6. The primary SFC uses the links 1-3 and 3-6 with the VNFs hosted at node 3. A backup SFC is also necessary to achieve the required availability and uses links 1-2 and 2-6 with the backup VNFs hosted at node 2.

Note that even considering the optional provisioning of backup resources, link 2-6 and node 2 will have twice their resources occupied, i.e., one for *SFC1* and one for *SFC2*. Given that their primary SFCs do not share the use of any link or node, one way to improve the resource efficiency, in this example, would be to share the backup resources assigned in link 2-6 and node 2 between the two SFCs. In this work, we investigate how to exploit the intuition of shared backup resources for the provisioning of SFCs.

III. SFC Provisioning with Optional Backup, Shared Path and Shared Function (OBSPSF)

This work proposes a SFC provisioning strategy called OBSPSF. The strategy aims at reducing the backup resource overhead of availability-constrained SFCs by *(i)* provisioning backup resources only when a single SFC cannot satisfy the availability requirements and *(ii)* sharing connectivity and processing backup resources among SFCs that have their primary resources disjoint (i.e., their primary resources are not deployed over the same nodes and links).

Upon the arrival of an SFC request, the strategy tries to find an SFC realization that satisfies the connectivity, processing, and availability requirements of the request. If multiple solutions can satisfy all these requirements, OBSPSF selects the one with the lowest cost (the cost computation will be discussed in detail in Sec. III-A). If no solution using a single SFC can satisfy the availability requirements of the SFC request, the strategy tries to find a pair of node-disjoint SFCs to serve as primary and backup. By provisioning backup resources only when the provisioning of the primary SFC is not sufficient to fulfill the SFC availability requirements, the strategy can improve resource efficiency. Moreover, when backup resources are necessary, OBSPSF encourages the sharing of resources by trying to find primary-backup SFC pairs that allow resources to be shared.

TABLE I: Notations

Notation	Description
$G(N, L)$	Network topology graph with $\|N\|$ nodes and $\|L\|$ links
$N^{DC} \subset N$	Set of network nodes that are colocated with a DC
p_n^{dc}	The number of processing units available at node $n \in N^{DC}$
c^{lnk}	Number of connectivity units at each link
S_r^{bkp}	Set of SFCs using the same connectivity/processing resource r
c_l^{bkp}	The set of connectivity units used as backup at link $l \in L$
F_n^{bkp}	The set of VNFs used as backup at node $n \in N^{DC}$
s	An SFC request to be provisioned
c_s^{sfc}	The number of connectivity resources required by SFC s
F_s^{sfc}	The set of VNFs required by SFC s
p_f^{sfc}	The number of processing resources required by a VNF $f \in F_s^{sfc}$
a_s^{sfc}	Minimum end-to-end availability required by SFC s
P_s	The set of single paths for SFC s
Q_s	The set of primary/backup path-pairs for SFC s
c_p^{path}	The number of available connectivity resources across path p
p_p^{path}	The number of available processing resources across DCs in path p
m_p	The maximum link load across all links in path $p \in P_s \vee p \in Q_s$
a_p^{sfc}	The availability obtained for SFC using $p \in P_s$
$a_{p,b}^{pair}$	The availability obtained by a pair of SFCs using $p, b \in Q_s$

Table I shows the notation adopted in this work, and we describe the core steps of the OBSPSF strategy in Alg. 1. The heuristic assumes to know all the input components described in the problem definition (Sec. II). The heuristic also assumes the existence of a list (P_s) computed offline with k-shortest paths between source and destination of the SFC request s. Moreover, the heuristic assumes the existence of a list (Q_s) with primary/backup node-disjoint path-pairs between source and destination of the SFC request s. The heuristic computes the availability using the generalized availability formulation introduced in [8] and presented in (1). The SFC availability (a_{sfc}) is computed based on the availability of each node and link traversed by the paths used by the primary and backup SFC($paths$). We apply (1) to computed the availability of a single SFC p (a_p^{path}) as well as the availability obtained by a pair of SFCs ($a_{p,b}^{path}$).

$$a_{sfc} = 1 - \prod_{p \in paths} ((1 - \prod_{n \in p^{nodes}} a_n^{node} \prod_{l \in p^{links}} a_l^{lnk})) \quad (1)$$

Alg. 1 works as follows. Lines 1-2 initialize support variables for the heuristic. The algorithm checks which paths are candidate for the SFC realization by verifying if the number of connectivity and processing resources available satisfies the SFC requirements, and if the cost of such a path is lower than the lowest one found so far (lines 4-5). If these conditions are satisfied, the path and its cost are saved in the support variables (lines 6-7). After traversing all the paths, the algorithm returns the path with the lowest cost, if a path was found (lines 10-12).

When no SFC is found satisfying all the requirements, it might be due to the fact that the availability offered by a single SFC is not enough to meet the required one. Therefore, the heuristic tries to find a pair of SFCs from Q_s (line 13) that can

Algorithm 1: Heuristic for the Optional Backup with Shared Path and Shared Function

```
Data: $G(N, L)$, $s$, $P_s$, $Q_s$
Result: Primary and backup resources for the request.
1  $minCost = max()$;
2  $bestSolPri = bestSolBkp = \varnothing$;
3  for $w \in P_s$ do
4      if $c_w^{path} \geq c_s^{sfc} \wedge p_w^{path} \geq \sum_{f \in F_s^{sfc}} p_f^{sfc}$
5      $\wedge a_w^{path} \geq a_s^{sfc} \wedge minCost > cost(s, w, \varnothing)$ then
6          $bestSolPri = w$;
7          $minCost = cost(s, w, \varnothing)$
8      end
9  end
10 if $bestSolPri \neq \varnothing$ then
11     return $bestSolPri, \varnothing$;
12 end
13 for $\{w, b\} \in Q_s$ do
14     if $c_w^{path} \geq c_s^{sfc} \wedge p_w^{path} \geq \sum_{f \in F_s^{sfc}} p_f^{sfc}$
15     $\wedge c_b^{path} \geq c_s^{sfc} \wedge p_b^{path} \geq \sum_{f \in F_s^{sfc}} p_f^{sfc}$
16     $\wedge a_{w,b}^{pair} \geq a_s^{sfc} \wedge minCost > cost(s, w, b)$ then
17         $bestSolPri = w$;
18         $bestSolBkp = b$;
19         $minCost = cost(s, w, b)$;
20     end
21 end
22 return $bestSolPri, bestSolBkp$;
```

satisfy the resource (lines 14-15) and availability requirements of the chain, and has the lowest cost so far (line 16). If the path pair satisfy all these requirements, it is stored in the support variables. Finally, the algorithm returns the solution when a solution is found, or empty otherwise.

Once a single or a pair of SFCs is found, the connectivity and processing resources are assigned. The assignment of primary resources follow a first-fit approach, i.e., choose the first connectivity unit(s) available at each link in the path and the first DC along the path with enough processing resources. When backup resources are necessary, the assignment is performed prioritizing the maximization of sharing and according to the cost computation function (detailed in the following section).

A. Cost Computation

Alg. 1 uses the *cost* function responsible for computing the cost associated with the provisioning solution of an SFC based on three components. The first component is a load balancing factor that weights the cost by the load observed in the path, assisting the heuristic to avoid bottlenecks. The second and third components account for the monetary cost of the resources occupied by the SFC. For these components, a factor β relates the cost of a connectivity unit to the cost of a processing unit [9]. The primary SFC cost accounts for the number of connectivity and processing units necessary to its provisioning. The backup SFC cost is similar to the primary one, with the difference being on the cost of the

Algorithm 2: Cost Computation

```
1  Function cost(s, w, b):
       Data: An SFC request ($s$), a primary SFC ($w$) and
             an optional backup SFC ($b$)
       Result: Cost associated with the primary/backup
               paths
2      $tCost$ = $pCost$ = 0;
3      $cCost$ = $c_s^{sfc} \times |w|$;
4      for $f \in F_s^{sfc}$ do
5          $pCost$ = $pCost$ + $p_f^{sfc}$;
6      end
7      $tCost$ = $m_w/c^{lnk} \times cCost + cCost + \beta \times pCost$;
8      if $b = \varnothing$ then
9          return $tCost$;
10     else
11         for each $c_s^{sfc}$, $l \in b$ do
12             shared = 1;
13             for $c \in c_l^{bkp}$ do
14                 if $disjoint(w,p) \forall p \in S_c^{bkp}$ then
15                     shared = max(shared, $|S_c^{bkp}| + 1$);
16                 end
17             end
18             $cCost$ = $cCost$ + $1/shared$;
19         end
20         $excl = \varnothing$;
21         for $f \in F_s^{sfc}$ do
22             for $n \in b \mid n \subseteq N^{DC} \wedge n \nsubseteq excl$ do
23                 $shared = 1$;
24                 for $g \in F_n^{bkp} \mid g = f$ do
25                     if $disjoint(w,p) \forall p \in S_g^{bkp}$ then
26                         $shared$ = $|S_g^{bkp}| + 1$;
27                         break;
28                     end
29                 end
30                 if $shared > 1 \vee p_n^{dc} \geq p_f^{sfc}$ then
31                     $pCost$ = $pCost$ + $p_f^{sfc}/shared$;
32                     break;
33                 else
34                     $excl$ = $excl \cup \{n\}$;
35                 end
36             end
37         end
38         $tCost$ =
             $tCost + m_b/c^{lnk} \times cCost + cCost + \beta \times pCost$;
39     end
40     return $tCost$;
```

shared resources. For the resources that can be shared, their cost is split among all the SFCs sharing them at the moment of the provisioning. By considering these three components, the *cost* function summarizes the overall cost of a candidate provisioning solution, facilitating the selection mechanism in Alg. 1.

Alg. 2 presents the cost calculation of a provisioning solution for single or primary/backup SFCs. First, the algorithm initializes support variables (line 2). Since the cost of primary resources does not depend on sharing, its computation is straightforward. The connectivity cost of the primary SFC is the computing of the number of connectivity units multiplied by the number of links traversed (line 3). The processing cost of the primary SFC is the number of processing units required by all the VNFs of the SFC (lines 4-6). The total cost of the primary SFC is the sum of the three factors, i.e., load balancing, connectivity, and processing resources (line 7). If the solution evaluated has no backup SFC, the algorithm returns the cost of the primary one (lines 8-10).

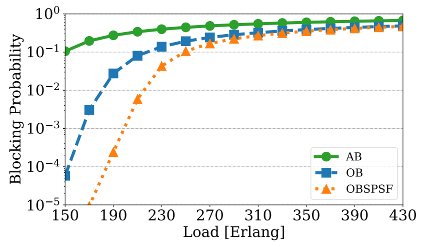
If the solution has a backup SFC, its cost computation involves a more elaborated procedure (lines 10-40). First, the sharing of connectivity resources is evaluated (lines 11-19). For each connectivity unit requested by the SFC, and for all the links traversed by the backup SFC, the algorithm counts how many SFCs will share a particular connectivity resource, and divides the cost among them. Next, we calculate the cost of backup processing resources (lines 20-37). For each VNF from F_s^{sfc}, the algorithm checks all DCs traversed by the backup SFC and finds backup VNFs already provisioned that can be shared, respecting the order and type of VNFs in the SFC. Finally, the total cost comprised of primary and backup SFCs is computed and returned. During the resource assignment a similar procedure can be used to find which resources should be assigned. When an SFC leaves the network, shared resources are only released if no other SFC is using it.

B. Complexity Analysis

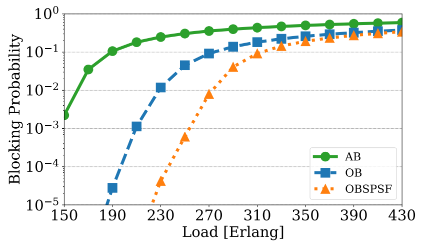
The worst case computational complexity of Alg. 1 accounting also for the *cost* function complexity, is denoted as $O\Big(|P_s|\,\left|F_s^{sfc}\right| + |Q_s|\,(\left|F_s^{sfc}\right| + |L|\,c_s^{sfc}c^{lnk}\left|S_r^{bkp}\right| + \left|F_s^{sfc}\right|$ $|N|\left|F_n^{bkp}\right|\left|S_r^{bkp}\right|)\Big)$. The first term accounts for the complexity of finding a single SFC, i.e., iterate over P_s. The second term accounts for the complexity of finding and calculating the cost of a primary and a backup SFCs, i.e., iterating over Q_s. We consider that, in the worst case, the paths traverse all links (L) and all nodes (N) of the topology under exam.

IV. Evaluation

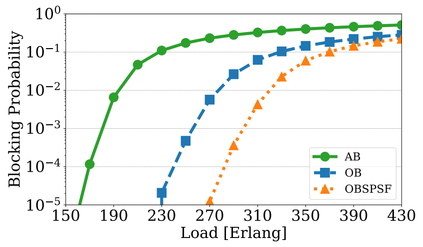
The performance of the OBSPSF is assessed through simulation, and compared with two other state-of-the-art strategies for availability-guaranteed SFC provisioning. The first benchmark is called Always Backup (AB), an availability-unaware strategy that provisions backup resources regardless of the availability requirements of the SFC. The second benchmark is called Optional Backup (OB), inspired by [5], which provisions a backup SFC only if the primary one is not sufficient to fulfill the availability requirements, and has no backup sharing capabilities. All three strategies tested (the two benchmarks and OBSPSF) only accept an SFC request if its availability requirements are satisfied. In the following, the simulation setup is described, and the simulation results are presented.



(a) θ=0.8

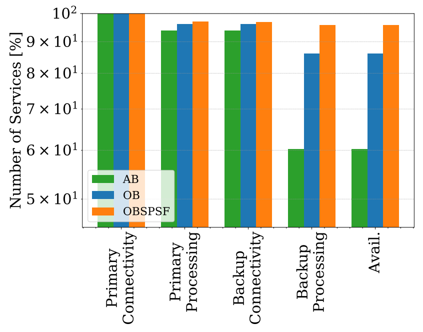


(b) θ=1.0

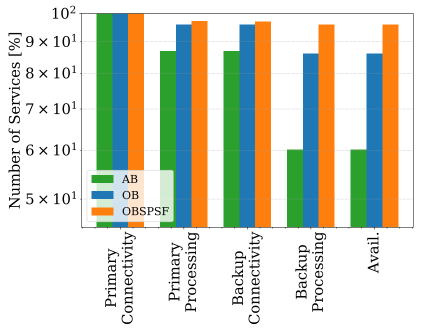


(c) θ=1.2

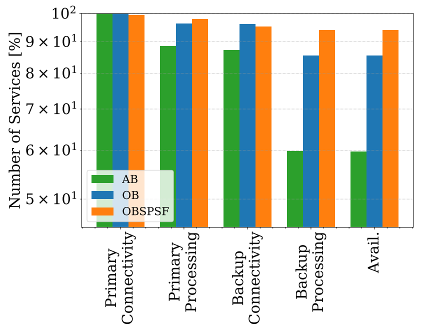
Fig. 2: Blocking Probability



(a) θ=0.8, load=230



(b) θ=1.0, load=290



(c) θ=1.2, load=350

Fig. 3: Number of SFC request requirements fulfilled for each dimensioning factor. Load selected for OBSPSF blocking probability around 5%.

A. Experimental Setup

For the simulations carried out in this paper, we considered the SmallNet topology illustrated in Fig. 1 with 10 nodes, of which 4 are DCs, and 22 links. At the beginning of the simulation, 20 shortest paths are computed for each node pair, i.e., $|P_s|$=20. For each shortest path, up to 10 node-disjoint paths are computed, i.e., $|Q_s| \leq 200$. We assume the availability of each node as 0.9999, the availability of each link as 0.99. Each network link has 80 connectivity units. The number of processing units at each DC is calculated according to (2), where p_n^{dc} is the number of processing units at DC node n computed as a function of the nodal degree (d_n), the number of connectivity units at each link (c^{lnk}), the average number of processing units per VNF (δ) and a dimensioning factor (θ). We evaluate three values of θ: $\theta = 0.8$ representing the case where processing resources are under-dimensioned; $\theta = 1.0$ representing the balanced infrastructure; and $\theta = 1.2$ representing the case where processing resources are over-dimensioned. For each θ value, we focus on the load that offers the blocking probability in the range between zero and 10%.

$$p_n^{dc} = d_n \times c^{lnk} \times \delta \times \theta \tag{2}$$

The holding time of each SFC is exponentially distributed with an average value of 60 time units. The arrival rate follows a Poisson distribution, where the mean time between arrivals varies according to the load value. We assume that each VNF requires one processing unit and that the SFC requires one connectivity unit. The source of an SFC is uniformly selected among the non-DC nodes, while destination is uniformly selected among the DC nodes. The set of VNFs composing the SFC is uniformly selected from the SFC types in Table II [1]. Note that the chains in Table II do not include the end service point, e.g., web server, therefore the assumption that the destination node must be a DC. The availability requirement of each SFC is uniformly selected between 0.95 and 0.999. For each load value, we perform 20 simulation runs, with 300,000 SFC request arrivals per run. The results obtained have a confidence value not higher than 6.5% of the blocking probability for 95% confidence level.

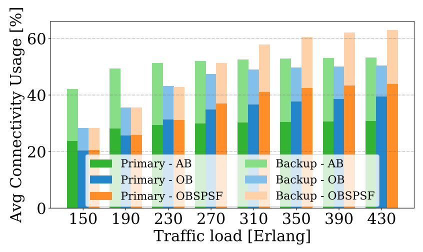
TABLE II: List of SFCs [1].

SFC	**Chained VNFs**
Web Service	NAT-FW-TM-WOC-IDPS
VoIP	NAT-FW-TM-FW-NAT
Video Conference	NAT-FW-TM-VOC-IDPS
Cloud Gaming	NAT-FW-VOC-WOC-IDPS
Augmented Reality	NAT-FW-TM-WOC-VOC

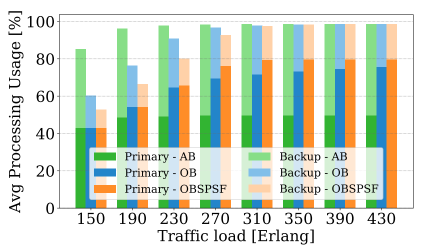
Network Address Translator (NAT), Firewall (FW), Traffic Monitor (TM), WAN Optimization Controller (WOC), Intrusion Detection Prevention System (IDPS) and Video Optimization Controller (VOS)

B. Numerical Results

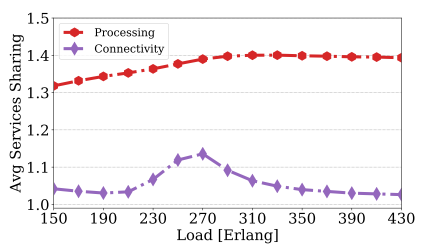
Fig. 2 shows the blocking probability for the three dimensioning factor values. Our strategy achieves a drastic reduction in blocking probability across all the load conditions and dimensioning factors, especially for blocking probabilities lower than 10%. Compared to AB, our strategy drastically reduces the blocking probability, not blocking any SFC request while AB blocks around 10% of the requests. Compared to OB, for load conditions where OB blocks around 1% of the SFC requests, our strategy reduces the blocking probability



(a) Connectivity usage



(b) Processing usage



(c) Number of services sharing a backup resource

Fig. 4: Average resource usage and number of services sharing backup resources for θ=1.0.

by around two orders of magnitude. These results show the benefits of combining availability-aware optional backup provisioning with the sharing of backup resources.

Fig. 3 shows the percentage of SFC requests that have their requirements fulfilled during provisioning and helps understanding the reasons why requests are blocked. All the strategies evaluated in this work check the requirements represented in the x axis of Fig. 3 sequentially. Note that there is a general trend of observing drops related to the lack of enough processing units. For instance, AB observes the most significant drops when checking available processing units, with backup processing units being the critical resource missing. OB presents a similar behavior, but with more services finding enough processing units. Finally, note that OBSPSF observes a slight drop when finding primary processing units. However, there is a high chance of finding backup units whenever primary resources are found. These values show that the sharing of backup resources effectively enables the accommodation of more SFCs.

Fig. 4 shows detailed resource usage percentages as well as the degree of shared resources by OBSPSF. Fig. 4a shows that, since the OBSPSF can accommodate more services, its connectivity usage is higher than the other strategies, demonstrating the higher resource efficiency of the proposed strategy. Fig. 4b shows that AB uses almost 100% of the processing resources already in very low loads, explaining the lack of processing resources observed in Fig. 3. Meanwhile, OBSPSF uses processing resources efficiently. For instance, in the highest load, OBSPSF assigns around 80% of the processing resources for use in the primary SFCs, using only the remaining 20% for backup. In similar load conditions, OB uses more than 25% of the processing resources for backup.

Finally, Fig. 4c shows the average number of services sharing a resource. Connectivity resources show a lower degree of sharing than processing resources. It is less likely to find connectivity units that can be shared because the network links have a much lower number of resources than DCs. For instance, in this work, we assume 80 connectivity units per link and at least 400 processing units per DC. Moreover, we observe an increase in connectivity sharing in the load range 230-310. This behavior is expected because of two aspects observed on medium load conditions: *(i)* for a connectivity unit to be shared, there should be enough path diversity in the network, i.e., enough SFCs so that one with node-disjoint primary path can be found; and *(ii)* there should be enough free resources so that SFC requests have enough resources for their primary paths. These aspects are not observed for processing units due to the high number of units within the same DC, resulting in a higher likelihood of sharing.

V. Conclusion

In this paper, we propose Optional Backup with Shared Path and Shared Function (OBSPSF), an availability-aware strategy that provisions availability guaranteed SFC, where the backup connectivity and VNF resources are shared when possible. Our simulation results show that the proposed strategy can accommodate an orders-of-magnitude-higher number of SFC requests due to its improved resource efficiency. A detailed analysis shows that the proportion of resources assigned for backup is much lower than the ones achieved by benchmark strategies taken from the literature. As future work, we plan to investigate the efficiency of the proposed algorithm for SFCs with a varying number of required VNFs.

References

[1] A. Hmaity *et al.*, "Latency- and capacity-aware placement of chained virtual network functions in FMC metro networks," *Optical Switching and Networking*, vol. 35, 2020, DOI: 10.1016/j.osn.2019.100536.

[2] G. Mirjalily and Z. Luo, "Optimal network function virtualization and service function chaining: A survey," *Chinese Journal of Electronics*, vol. 27, no. 4, pp. 704–717, 2018, DOI:10.1049/cje.2018.05.008.

[3] B. Han *et al.*, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, Feb 2015, DOI:10.1109/MCOM.2015.7045396.

[4] L. Qu *et al.*, "Reliability-Aware Service Chaining in Carrier-Grade Softwarized Networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 558–573, 2018, DOI: 10.1109/JSAC.2018.2815338.

[5] J. Kong *et al.*, "Guaranteed-Availability Network Function Virtualization with Network Protection and VNF Replication," in *Global Communications Conference (GLOBECOM)*, Singapore, Dec. 2017, pp. 1–6, DOI: 10.1109/GLOCOM.2017.8254730.

[6] ——, "Guaranteed-availability network function virtualization in inter-datacenter networks," in *Optical Fiber Communications Conference and Exposition (OFC)*, San Diego, CA, USA, Mar. 2018, DOI: 10.1364/OFC.2018.W1D.1.

[7] J. Fan, *et al.*, "GREP: Guaranteeing reliability with enhanced protection in NFV," in *Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddlebox)*, London, United Kingdom, Aug. 2015, pp. 13–18, DOI: 10.1145/2785989.2786000.

[8] J. Fan *et al.*, "Availability-aware mapping of service function chains," in *Conference on Computer Communications (INFOCOM)*, Atlanta, GA, USA, May 2017, pp. 1–9, DOI: 10.1109/INFOCOM.2017.8057153.

[9] C. Natalino *et al.*, "Dimensioning optical clouds with shared-path shared-computing (SPSC) protection," in *International Conference on High Performance Switching and Routing (HPSR)*, Budapest, Hungary, Jun. 2016, DOI: 10.1109/HPSR.2015.7483085.