This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2019.2959818, IEEE Transactions on Network and Service Management

Reliability Aware Service Placement Using a Viterbi-based Algorithm

Mohammad Karimzadeh-Farshbafan, Vahid Shah-Mansour, *Member, IEEE,* and Dusit Niyato, *Fellow, IEEE*

***Abstract*—Network function virtualization (NFV) is referred to as the deployment of software functions running on commodity servers, instead of hardware middleboxes. It is an inevitable technology for agile service provisioning in next-generation telecommunication networks. A service is defined as a chain of software functions, named virtual network functions (VNFs), where each VNF can be placed on different host servers. The task of assigning the VNFs to the host servers is called service placement. A significant challenge in service placement is meeting the reliability requirement of a service. In the literature, the problem of service placement and providing the required reliability level are considered separately. First, the main server is selected, and then, the backup servers are deployed to meet the reliability requirement of the service. In this paper, we consolidate these two steps and perform them jointly and simultaneously. We consider a multi-infrastructure network provider (InP) environment where InPs offer general purpose commodity servers with different reliability levels. Then, we propose a programming problem for main and backup server selection jointly minimizing the cost of resources of the InPs and maximizing the reliability of the service. We reformulate this problem as a mixed integer convex programming (MICP) problem. Since MICPs are known to be NP-hard in general, we propose a polynomial time sub-optimal algorithm named Viterbi-based Reliable Service Placement (VRSP). Using numerical evaluations, we investigate the performance of the proposed algorithm compared to the optimal solution resulting from the MICP model and also with three heuristic algorithms.**

***Index Terms*—Network Function Virtualization, reliable service placement, mixed integer convex programming, Viterbi algorithm.**

I. Introduction

One of the important obstacles for the enterprise networks to provide agile services is the dependency on network hardware namely hardware middleboxes [1]. Adding a new functionality to such networks requires the deployment of new hardware which makes agile service provisioning difficult [2]. To obviate the limitation of middleboxes, network function virtualization (NFV) is used [3], [4]. In the NFV paradigm, the hardware middleboxes are replaced by the modules of software named virtual network functions (VNFs) running on commodity servers, e.g., x86-based systems. It is worth noting that the software implementation of network functions can achieve hardware middleboxes performance [5]. For providing a specific service, a set of VNFs are chained. This chain of VNFs is named service function chain (SFC) [6]. In an SFC, some functions can be implemented using VNFs, and some can be implemented as traditional physical network functions (PNFs). However, throughout the paper, we consider that the functions of an SFC are implemented using VNFs. Nevertheless, the extension to the case where a service is composed of VNFs and PNFs is straightforward. Similar to PNFs, VNFs of a service may have location constraints. However, it is easier to move VNFs than PNFs. This possibility of dynamic placement of VNFs creates a new opportunity for reducing the OPEX and makes agile service provisioning feasible. In our network model, there are three main components:

- **Infrastructure network provider** (InP) is the owner of the commodity servers. InP also provides network infrastructure (i.e., links) for connection of the VNFs.
- **Services** are requested by the network users and require special service level agreements (SLAs) (e.g., the reliability of service or end to end delay).
- **Network Operator** (NO) is responsible for providing services to users. For each service request, NO efficiently places and schedules the VNFs of the chain onto InP [1].

An important task of the NO is the service placement in which it seeks to find InPs on which it can place the VNFs of the services, considering incoming services and resources of InPs. Service placement is a two-stage problem including node and link mapping. Each VNF is characterized by a type (i.e., storage, computing or networking), and it has to be placed on a node of the InP that meets the VNF's type. Due to the limitation of InP resources, the NO should be able to place SFCs in a way that maximizes the number of admitted services while minimizing the total placement cost. Two main components of the placement cost are the operational cost of the servers running the VNFs and the networking cost of the links between the VNFs.

The NO should meet the service SLA in the service placement procedure. The reliability of the service, as a main factor of the SLA, is defined as the probability that the SFC is seamlessly running at a certain time. An SFC is seamlessly running if all its constituent VNFs run without failure. Nevertheless, the reliability of a service depends on the reliability of the constituent VNFs. The reliability of each VNF depends on the reliability of the commodity server, which executes the VNF. In this regard, if only one of the host servers of the constituent VNFs of a service fails, the respective VNF, and the service would be disrupted. The software implementation of network functions can increase the failure probability of a service compared to using middleboxes, and the NO should consider this effect in the placement. Therefore, the NO should allocate the servers with appropriate reliability level to the VNFs of an SFC to meet the required

M. Karimzadeh-Farshbafan and V. Shah-Mansour are with the School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran (e-mail: vmansouri@ut.ac.ir).

D. Niyato is with School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore.

reliability. If one instance of a VNF cannot meet the required reliability level, multiple instances of a VNF are placed as backups to reach that reliability level. Intuitively, for a service with a high-reliability requirement, the NO uses the servers with low failure probability and vice versa. However, in most scenarios, the assignment of one server to the VNFs of a service is not sufficient for meeting the required reliability. As a result, to meet the reliability requirement, NOs usually employ hot backups. This means that the NO assigns more than one server for all or some of the VNFs in the SFC. The first server is called the main server, and the next servers are called the backup servers. NOs can follow two approaches for reliability-aware service placement. In the first one, the main server is selected, and if the reliability is not met, the backup servers are deployed after. The second approach considers the main and backup server assignment in one problem.

To the best of our knowledge, there is no comprehensive work considering the main and backup server selection jointly for reliability-aware service placement. The simultaneous selection of the main and backup servers can reduce the placement cost as well as the required number of backups for meeting the reliability requirement. Also, this placement method can increase the number of admitted services. In such a scenario, assigning a high-reliable server as the main server to a VNF can obviate the need for the backup server. On the other hand, when a backup server is crucial for the VNF, we can assign a low-reliable server as the backup server of such VNF. Obtaining an optimal solution from the perspective of placement cost and admission ratio is only possible when the main and backup servers selection are considered simultaneously. Therefore, we propose a novel programming problem for reliability-aware service placement in which the main and backup server assignment is considered simultaneously. The contributions of this paper are summarized as follows:

- We consider a multi-infrastructure (multi-InP) provider scenario in which each InP offers servers with certain reliability to the NO.
- We formulate an optimization problem that considers main and backup server assignment jointly.
- We reformulate the original optimization problem as a mixed integer convex programming (MICP) for which there exist well-suited algorithms [7], [8].
- We introduce a sub-optimal algorithm named Viterbi-based reliable service placement (VRSP) algorithm for solving the optimization problem. In the proposed algorithm, the decision metric is defined based on the sum of the placement cost and a penalty term for violating the required reliability.

The rest of the paper is organized as follows. The existing methods for reliability-aware service placement in NFV are reviewed in Section II. We present the system model and the proposed optimization problem in Section III. In Section IV, we reformulate the problem in a more tractable form. We introduce our heuristic VRSP algorithm for solving the optimization problem in Section V. In Section VI, we investigate the performance of the proposed algorithm compared with the optimal solution and three existing methods for backup assignment in NFV. Paper is concluded in Section VII.

II. Related Work

In this section, we first discuss service placement algorithms without considering reliability requirements. Then, we consider reliability-aware service placement approaches.

In [9], the service placement problem with the purpose of energy and traffic cost minimization and preventing resource fragmentation in the servers is considered. In [10], the service placement problem is considered in a way that the cost of using servers and links is minimized and the requested delay of services is met. In [11], the service placement problem with a cost function, including deployment, resource, traffic, delay, and resource fragmentation costs is considered. In [12], the authors propose a migration policy that determines when and where to use migration. In [13], [14], a dynamic market mechanism design for on-demand SFC provisioning and pricing in the NFV market is studied. The authors in [15] model the selfish and competitive behavior of users when an atomic weighted congestion game is used. In [16], a partitioning game is used for splitting an SFC in a set of partitions executed as virtual machines/containers in the appropriate servers of a cloud environment satisfying server affinity, collocation, and latency constraints. In [17], [18], service placement is considered for the data plain of NFV-enabled cellular networks. In [19], service placement for both data and control plane in evolved packet core (EPC) considering stringent delay budgets among cellular core components is studied. In [20], service placement and scheduling in the radio access network (RAN) of the mobile virtual network operator are considered.

Existing placement techniques consider backup server assignment to VNFs independently from the main server selection phase. In fact, they assign the main server first and then try to add backup servers to meet the reliability requirement of the services [21]–[24]. Also, most of these techniques deploy the backup without considering the reliability of the InPs hardware. However, authors in [25] consider the effect of the InP's reliability in their model. In [21], [22] the backup server selection is only considered after the main server has been selected. The approach in [21] treats the selection of VNFs for backup independently from the backup placement. This means that first VNF of the SFC which does not have a backup server, is selected. Then, the backup server allocation to the selected VNF is performed. These two steps should be considered simultaneously for finding the optimal solution. The proposed method in [21] leads to a solution in which all VNFs of an SFC have a backup server independent of their reliability requirement. This results in unnecessary cost.

In [22], the VNFs that have been placed on unreliable servers are selected to have a backup server. In [23], a new iterative cost-effective redundancy algorithm named CERA is proposed in which VNF selection for backup and backup placement are combined. At each iteration, the proposed method determines the VNFs that should have a backup server and also the server for hosting the selected VNFs using a metric named cost importance measure. This metric for the i^{th} VNF of an SFC is defined as an increase in the reliability value if the i^{th} VNF has a backup server divided by the cost

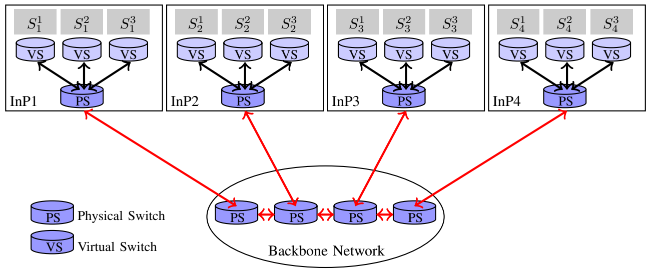


Fig. 1. An example of the Multi-Infrastructure Network Provider scenario.

of the backup server for this VNF. In [24], virtual backup assignment for middleboxes is considered. The method in [24] allocates only a few backup servers where each of them is a backup for the multiple middleboxes. The authors in [22]–[24] considered the backup assignment separately from the main server selection. The authors in [25], considered iterative backup selection with a routing procedure and endeavored to maximize link utilization while providing required reliability and delay. They have not investigated the main and backup server selection simultaneously. Also, the effect of link and server cost on the placement problem has not been considered. Therefore, the authors in [25] did not discriminate between the cost of using different servers according to their reliability. In [26], an integer nonlinear programming approach is proposed to determine the minimum number of backup servers for an incoming service. The authors also design an online heuristic algorithm for maximizing the number of service requests that can be served. Similar to [21]–[25], the most important shortcoming of [26] is that the selection of the main and backup servers is considered separately, which leads to a sub-optimal solution. In [27], for dynamic reliability-aware service placement only using the main server, a deep reinforcement learning (Deep-RL) method is proposed.

III. System Model

We consider a scenario in which a NO aims to deliver agile services using NFV. There are multiple InPs with commodity servers. Each InP offers servers with different amounts of resources and different levels of reliability. In this regard, one can cluster the servers into some virtual InPs where the servers of a virtual InP have the same reliability level. Figure 1 shows an example of the introduced multi-InP scenario, which includes four InPs. It is worth noting that each InP is a physical InP which has three servers, a virtual switch (VS) on each server and a physical switch (PS). The considered VSs are responsible for traffic routing to and from different virtual machines (VMs) in the respective server. The considered PSs are responsible for routing the traffic between the servers of the respective InP. Also, the PS in each InP and the considered PSs in the backbone network are responsible for routing the traffic between two servers in different InPs.

A. Infrastructure Network Providers

Let P denote the set of InPs, and S_i denote the set of servers of InP i. The number of InPs is $|P|$ and the number of servers of i^{th} InP is $|S_i|$. We represent the entire network of InPs as an undirected graph $G = (S, L)$, where S indicates the set of servers and L is the set of the links between the servers as

$$S = \{S_i^n \mid n \in \{1, 2, \ldots, |S_i|\}, i \in \{1, 2, \ldots, |P|\}\}, \quad (1)$$

$$L = \{L_{i,j}^{n,m} \mid n \in \{1, 2, \ldots, |S_i|\},\ m \in \{1, 2, \ldots, |S_j|\},\\ i, j \in \{1, 2, \ldots, |P|\}\}, \quad (2)$$

where S_i^n indicates the n^{th} server of the i^{th} InP and $L_{i,j}^{n,m}$ indicates the logical link between the n^{th} server of the i^{th} InP and the m^{th} server of the j^{th} InP. Each logical link between two servers represents a physical path in the network of the InPs which is composed of one or more physical links. In the rest of the paper, we use link instead of logical link. The computational resource of the n^{th} server in the i^{th} InP is denoted by R_i^n. The bandwidth of the link between the n^{th} server in the i^{th} InP and the m^{th} server in the j^{th} InP is denoted by $B_{i,j}^{n,m}$. The unit cost of using servers of the i^{th} InP is denoted by C_i and the unit cost of using the link between the n^{th} server of the i^{th} InP and the m^{th} server of the j^{th} InP is denoted by $C_{i,j}^{n,m}$. The cost of using servers and links are defined in the monetary form. The cost of using inter-InPs and intra-InP links can be different in practice. The unit cost of using a server increases with enhancement in the reliability of the server. As a result, the cost of using a server is determined according to its reliability. Without loss of generality, let v_i indicate the failure probability of the servers of the i^{th} InP. We assume that decreasing the failure probability marginally close to zero exponentially increases the cost of each InP's servers as

$$C_i = \alpha e^{\beta(v_{\text{Base}} - v_i)},\ i = 1, \ldots, |P|, \quad (3)$$

where α and β are design parameters for justifying the order of the C_i compared to the other variables of the problem. Also, v_{Base} is the highest acceptable failure probability, and the failure probability of each InP should be lower than this threshold. We know that with an increase in the reliability value of a server, the involved hardware in the server will be more expensive, which leads to an increment of server cost. One way to determine the cost of the servers according to their reliability is to consider the downtime of each reliability value. For the servers with failure probabilities of $\{0.05, 0.03, 0.01\}$, the downtimes are $\{18.26, 10.96, 3.65\}$ days per year. Assume that the cost of the first server is 1, we can consider the costs of the second and third servers to be 1.66 and 5, respectively, according to their lower downtime. This range of values for the cost of the servers can be calculated using the introduced exponential model. By using this model the high-reliable servers become expensive, and as a result, the NO only uses the high-reliable servers in case of not assigning backup servers. Without this model, the resources of high-reliable servers can be wasted, which can lead to a reduction in the admission ratio and an increase in the placement cost. It is worth noting that NOs have complete information about the cost of using the resources of the different InPs as well as the reliability of each InP.

B. Characteristics of Service Requests

As mentioned before, a specific service is a chain of VNFs called SFC. We assume a slotted time structure in which the time is divided into equal slots. At the beginning of the t^{th} slot, we consider the service placement problem for the incoming services during the $(t-1)^{\text{th}}$ slot. Also, we assume that the admitted services at the beginning of the t^{th} slot leave the network during this slot and last up to a maximum of one slot. In practice, the services may last more than one slot. The scenario in which each service can last for more than one slot can be mapped to the scenario in which each service lasts up to one slot. This mapping can be done with one of the following solutions:

1) The admitted services at the $(t-1)^{\text{th}}$ slot which have not left the network will be considered in the service placement problem at the beginning of the t^{th} slot. It is worth noting that there will be enough resources for the placement of the remaining services. However, the placement of services which last for more than one slot can be changed in the next slot.
2) The allocated resources for the placement of the admitted services in the $(t-1)^{\text{th}}$ slot, which last more than one slot, will not be used for the service placement problem at the beginning of the t^{th} slot.

Let K denote the number of requested services in a slot which is a random variable. We assume that the k^{th} service's SFC includes U_k VNFs. We indicate the required bandwidth and the required resource of the u^{th} VNF of the k^{th} SFC with b_k and r_k^u, respectively. We consider only one resource type for requested services. The extension to a scenario with multiple resource types is considered at the end of this section. The decision variable of placing the u^{th} VNF of the k^{th} SFC in the n^{th} server of the i^{th} InP is indicated by $x_{i,k}^{n,u} \in \{0,1\}$.

C. Cost Function

The considered cost function is composed of two main components. The first component is the cost of using the servers denoted by ξ_s. We assume that the cost of using a server is linearly proportional to the amount of resource utilized. ξ_s is the summation of the multiplication of the required resource for the u^{th} VNF in the k^{th} service and the cost of using i^{th} InP, if the respective VNF is placed in the n^{th} server of the respective InP. ξ_s can be expressed as:

$$\xi_s = \sum_{k=1}^{K} \sum_{u=1}^{U_k} \sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \times r_k^u \times C_i. \quad (4)$$

The second component of the cost function is the cost of using links between servers which is denoted by ξ_l and is

$$\xi_l = \sum_{k=1}^{K} \sum_{u=1}^{U_k-1} \sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} \sum_{j=1}^{|P|} \sum_{m=1}^{|S_j|} x_{i,k}^{n,u} \times x_{j,k}^{m,u+1} \times b_k \times C_{i,j}^{n,m}, \quad (5)$$

where $x_{i,k}^{n,u} \times x_{j,k}^{m,u+1}$ is used to indicate the use of the link between the n^{th} server of the i^{th} InP and the m^{th} server of the j^{th} InP, for forwarding of the traffic between u^{th} and $(u+1)^{\text{th}}$ VNFs of k^{th} service. Note that even when two VNFs are placed in the same server (i.e., $n = m$ and $i = j$), they will still need a virtual link between them, and the cost of using this link must be considered. However, we assume that the cost of using a logical link between two servers is much higher than the cost of using a virtual link in a server. As a result, we do not consider the cost of using a virtual link in the placement cost and when ($n = m$ and $i = j$), then $C_{i,j}^{n,m} = 0$. Nevertheless, our model can be straightforwardly extended to the case where links within a server have non-zero cost. As seen in (5), this cost component is a nonlinear function of the binary decision variable, $x_{i,k}^{n,u}$.

The total cost function is sum of these components:

$$\xi_T = \xi_s + \xi_l. \quad (6)$$

D. Reliability Constraint

As mentioned, we would like to consider the service placement problem with the reliability requirement of requested services. As a result, one of the critical constraints for our problem is the reliability of the accomplished placement. If we indicate the failure probability of the k^{th} service with f_k and the maximum acceptable failure probability of this service with F_k, the reliability constraint is $f_k \leq F_k$.

We know that the failure probability of the k^{th} service, f_k, is a function of the binary decision variable, $x_{i,k}^{n,u}$. To obtain f_k, we should calculate the probability of being in the running state (i.e., not failed) for this service, q_k. A service is in the running state if all constituent VNFs of the service run without failure. As a result, we should determine the failure probability of a VNF as a function of the binary decision variable, $x_{i,k}^{n,u}$. Let f_k^u denote the failure probability of the u^{th} VNF of the k^{th} service. This probability is calculated by:

$$f_k^u = \prod_{i=1}^{|P|} \left(\prod_{n=1}^{|S_i|} \rho_{i,k}^{n,u} \right), \quad \rho_{i,k}^{n,u} = \begin{cases} v_i, & x_{i,k}^{n,u} = 1 \\ 1, & x_{i,k}^{n,u} = 0 \end{cases}, \quad (7)$$

where v_i is the failure probability of the i^{th} InP and $\rho_{i,k}^{n,u} = v_i$ when $x_{i,k}^{n,u} = 1$, else it is 1. According to (7), the failure probability of the u^{th} VNF in the k^{th} service is the multiplication of the failure probability of the VNF in all the InPs. Also, the failure probability of the VNF in each InP is the probability of failure of all the servers that the respective VNF is placed on them. We assume failure events in different InPs and also in different servers of an InP are independent. Now, we can calculate the probability of being in the running state for the k^{th} service, q_k, as the multiplication of the probabilities of all VNFs of the corresponding service running properly as

$$q_k = \prod_{u=1}^{U_k} (1 - f_k^u). \quad (8)$$

Finally, we can compute the failure probability of the k^{th} service as $f_k = 1 - q_k$.

E. Minimum Cost Service Placement

We formulate the optimization problem for the service placement in each slot as follows:

$$\min_{x_{i,k}^{n,u}} \quad \xi_T \quad (9)$$

$$\text{s. t.} \quad \sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \geq 1 \quad (10)$$

$$\prod_{u=1}^{U_k} \left(1 - \prod_{i=1}^{|P|} \left(\prod_{n=1}^{|S_i|} \rho_{i,k}^{n,u} \right) \right) \geq (1 - F_k) \quad (11)$$

$$\sum_{k=1}^{K} \sum_{u=1}^{U_k} x_{i,k}^{n,u} \times r_k^u \leq R_i^n \quad (12)$$

$$\sum_{k=1}^{K} \sum_{u=1}^{U_k-1} x_{i,k}^{n,u} \times x_{j,k}^{m,u+1} \times b_k \leq B_{i,j}^{n,m} \quad (13)$$

$$x_{i,k}^{n,u} \in \{0,1\} \quad (14)$$

$$i,j = 1,\ldots,|P|,\ n = 1,\ldots,|S_i|,\ m = 1,\ldots,|S_j|$$
$$k = 1,\ldots,K, u = 1,\ldots,U_k.$$

The constraint in (11) guarantees the reliability requirement of the k^{th} service. The constraint in (10) indicates that each VNF should be instantiated at least once. This constraint allows multiple placements of a VNF for meeting the reliability requirement. We assume that the range of the requested service levels matches the infrastructure capabilities. This ensures that the number of backup servers does not increase unboundedly. The constraints in (12)-(13) are used to make sure that the resource capacity of each server and the bandwidth capacity of each link are not violated. According to (13), we consider the service's SFC as a series of interconnected VNFs and ignore the branching SFCs.

we consider three bodies including NOs, InPs and services in this paper. Typically, the telecom NOs currently own the infrastructure. However, the market is moving towards infrastructure/network as a service model. We believe that the future model for telecom actual or virtual NOs would be towards the separation of InPs and NOs. Nevertheless, in a scenario in which the NO is the infrastructure owner, it still needs to optimize the placement cost. The latter can be considered as an operational expenditure (OPEX) in this case. As a result, the introduced optimization problem is still useful for reliability-aware service placement.

F. *Extension to Multiple Resource Types*

The optimization problem for minimum cost service placement in Section III-E, only considers one resource type for the VNFs. For the extension to a scenario in which each VNF requires multiple resource types, we consider $r_k^{u,t}$ as the t^{th} resource type requirement for the u^{th} VNF of the k^{th} service. Also, the resource capacity of each server is denoted by $R_i^{n,t}$ which indicates the t^{th} resource type capacity of the n^{th} server in the i^{th} InP. The cost of using servers, ξ_s, is revised as

$$\xi_s = \sum_{k=1}^{K} \sum_{u=1}^{U_k} \sum_{t=1}^{|T|} \sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \times r_k^{u,t} \times C_i, \quad (15)$$

where $|T|$ denotes the total number of resource types requested by the VNFs of services. The constraint in (12) is revised as

$$\sum_{k=1}^{K} \sum_{u=1}^{U_k} x_{i,k}^{n,u} \times r_k^{u,t} \leq R_i^{n,t}, \quad (16)$$
$$i = 1, \ldots, |P|, \; n = 1, \ldots, |S_i|, \; t = 1, \ldots, |T|.$$

By these modifications, the introduced optimization problem in (9)-(14), can be extended to a scenario in which each VNF requires multiple resource types.

IV. MICP Reformulation

In this section, we reformulate the optimization problem in (9)-(14) to a more tractable problem. The binary variable $x_{i,k}^{n,u}$ is used in a non-linear form in (13). Also, the reliability requirement constraint in (11) is a nonlinear function of $x_{i,k}^{n,u}$. This non-linearity is because f_k^u is a non-linear function of the binary variable $x_{i,k}^{n,u}$ in (7) and q_k is a non-linear function of f_k^u in (8). These non-linearities make the proposed optimization problem much more complex. We make a series of changes to convert the introduced optimization problem to an MICP for which there exist more efficient solvers.

First, we eliminate the non-linearity in (13). To do that, we introduce a new binary variable, $y_{i,j,k}^{n,m,u} \in \{0,1\}$ as

$$y_{i,j,k}^{n,m,u} = x_{i,k}^{n,u} \times x_{j,k}^{m,u+1}, \;\; 1 \leq u \leq U_k - 1, \;\; 1 \leq i,j \leq |P|,$$
$$1 \leq n \leq |S_i|, \;\; 1 \leq m \leq |S_j|, \;\; 1 \leq k \leq L. \quad (17)$$

Because the new variable should play the same role as $x_{i,k}^{n,u} \times x_{j,k}^{m,u+1}$ in the optimization problem, we add two new constraints as

$$y_{i,j,k}^{n,m,u} \leq x_{i,k}^{n,u}, \quad y_{i,j,k}^{n,m,u} \leq x_{j,k}^{m,u+1},$$
$$y_{i,j,k}^{n,m,u} \geq \left(x_{j,k}^{m,u+1} + x_{j,k}^{m,u+1} - 1\right). \quad (18)$$

For eliminating the non-linearity in (7), we replace f_k^u by a new continuous variable e_k^u throughout the problem, eliminate (7), and then introduce a new constraint as follows:

$$\prod_{i=1}^{|P|} \left(\prod_{n=1}^{|S_i|} \rho_{i,k}^{n,u} \right) \leq e_k^u \leq F_k, \quad (19)$$

It is worth noting that at the optimal point of the optimization problem, the left-hand side of the constraint in (19) is active which means that the left-hand side inequality becomes an equality. We employ the proof by contradiction to show this fact. Assume that at the optimal point, the left-hand side of (19) is less than e_k^u. In this case, the NO can increase the value of $\prod_{i=1}^{|P|} \left(\prod_{n=1}^{|S_i|} \rho_{i,k}^{n,u} \right)$ without violating any constraint. This leads to the reduction of the placement cost which leads to a point with lower objective than the optimal point. This is in contradiction with the optimality condition. As a result, at the optimal point, we have $e_k^u = \prod_{i=1}^{|P|} \left(\prod_{n=1}^{|S_i|} \rho_{i,k}^{n,u} \right)$. Now, we apply a logarithmic function to (19) and obtain

$$\ln(\rho_{i,k}^{n,u}) = \begin{cases} \ln(v_i), & x_{i,k}^{n,u} = 1 \\ 0, & x_{i,k}^{n,u} = 0 \end{cases},$$
$$\ln(e_k^u) \geq \sum_{i=1}^{|P|} \left(\sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \times \ln(v_i) \right). \quad (20)$$

The modified constraint in (20) is a linear function of binary random variable $x_{i,k}^{n,u}$ and also is convex with respect to variable e_k^u.

For eliminating the non-linearity in (8), we replace q_k with a new continuous variable c_k, eliminate (8), and introduce a new constraint as

$$0 \leq c_k \leq \prod_{u=1}^{U_k} (1 - e_k^u). \quad (21)$$

It is worth noting that in the optimal point of the optimization problem, the right-hand side inequality of (21) will be held in equality form. Now, we apply a logarithmic function to both sides of (21) and obtain

$$\ln(c_k) \leq \sum_{u=1}^{U_k} \ln(1 - e_k^u). \quad (22)$$

We approximate both sides of (22) using $\ln(1+x) \sim x$ for small x, and the constraint is changed as follows:

$$c_k + \sum_{u=1}^{U_k} e_k^u \leq 1. \quad (23)$$

This approximation is valid because the value of the reliability requirement is very close to 1. As a result, the desired value of c_k should be close to the reliability requirement and the value of e_k^u should be close to zero. This new constraint is a linear function of c_k and e_k^u. Now, we can eliminate the constraint in (11) and add two new constraints in (20) and (23) to the optimization problem. It is worth noting that for these modifications, we add two continuous decision variables, e_k^u and c_k, to the introduced optimization problem in (9)-(14). The modified version of the problem can be written as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2019.2959818, IEEE Transactions on Network and Service Management

$$\min_{x_{i,k}^{n,u}, y_{i,j,k}^{n,m,u}, e_k^u, c_k} \xi_T \tag{24}$$

$$\text{s. t. } \textstyle\sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \geq 1 \tag{25}$$

$$\ln(e_k^u) \geq \textstyle\sum_{i=1}^{|P|} \left(\sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \times \ln(v_i) \right) \tag{26}$$

$$c_k + \textstyle\sum_{u=1}^{U_k} e_k^u \leq 1 \tag{27}$$

$$\textstyle\sum_{k=1}^{K} \sum_{u=1}^{U_k} x_{i,k}^{n,u} \times r_k^u \leq R_i^n \tag{28}$$

$$\textstyle\sum_{k=1}^{K} \sum_{u=1}^{U_k - 1} y_{i,j,k}^{n,m,u} \times b_k \leq B_{i,j}^{n,m} \tag{29}$$

$$y_{i,j,k}^{n,m,u} \leq x_{i,k}^{n,u}, \quad y_{i,j,k}^{n,m,u} \leq x_{j,k}^{m,u+1} \tag{30}$$

$$y_{i,j,k}^{n,m,u} \geq \left(x_{j,k}^{m,u+1} + x_{j,k}^{m,u+1} - 1 \right) \tag{31}$$

$$x_{i,k}^{n,u}, y_{i,j,k}^{n,m,u} \in \{0,1\} \tag{32}$$

$$0 \leq e_k^u \leq F_k, \quad 0 \leq c_k \leq 1 - F_k \tag{33}$$

$$1 \leq i,j \leq |P|, \quad 1 \leq n \leq |S_i|,$$

$$1 \leq m \leq |S_j|, \quad 1 \leq k \leq K, \quad 1 \leq u \leq U_k,$$

where binary decision variable, $x_{i,k}^{n,u}$, is used in linear form. On the other hand, the decision variable, e_k^u, is used in a nonlinear form in (26). However, (26) is convex in inequality constraint with respect to e_k^u. As a result, the optimization problem in (24)-(33) has an MICP form.

V. The Proposed Algorithm

As mentioned in Section IV, the revised version of the optimization problem has an MICP form. However, the optimization problem in (24)-(33) is still computationally complex. Therefore, we propose a heuristic solution which achieves a sub-optimal solution with lower complexity. We would like to introduce a Viterbi-based algorithm named Viterbi-based Reliable Service Placement (VRSP) for solving this optimization problem. The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states (called the Viterbi path) that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models. A Viterbi path is defined as an example of a sequence for observed events. For example, for a sequence of received symbols in a wireless channel, the Viterbi algorithm is used to find a sequence (i. e., Viterbi path) with the maximum likelihood to the received symbols.

For finding the optimum path, the Viterbi algorithm first models the states of the problem and their transitions as a multistage graph. The number of stages is equal to the number of observed events. In each stage, one of the observed events, i.e., one of the received symbols in the wireless channel example, is considered. The state is defined as the possible events in each stage (i.e., possible options for the transmitted symbol in wireless channel example). In this model, each stage consists of all possible states. Also, for the transition between all pairs of the states in the consecutive stages, a transition cost is determined. This transition cost is defined according to the objective function of the optimization problem.

For each state of a stage, a path with minimum cost is selected as the survived path between the input paths to the respective state. It is worth noting that to determine the survived path of a state, the cumulative cost of input paths to the respective state is considered. In the last stage, each possible state has a survived path with a cost. The survived path of state with minimum cost is selected as a Viterbi path in the last stage. According to this description, we first determine state, stage and transition cost in our problem.

A. Viterbi based Reliable Service Placement Algorithm

We know that the number of requested services in each slot is K. Also, the k^{th} service in each slot is composed of a chain of U_k VNFs. For simplification of notations and presentations, we assume that the NO can assign at most one backup server for each VNF of a service. We shall notice that the solution can be easily extended to the case where a higher number of backup servers is needed. Using this assumption, the constraint in (10) is revised as $1 \leq \sum_{i=1}^{|P|} \sum_{n=1}^{|S_i|} x_{i,k}^{n,u} \leq 2$.

Let L_V denote the number of stages in the VRSP algorithm for each slot. L_V can be written as $L_V = \sum_{k=1}^{K} 2U_k$, where the coefficient 2 shows two server assignments including the main and backup server for each VNF. In each stage, selection of the main or backup server for a specific VNF in InPs is considered. For example, in the first stage, the selection of the *main server* for the first VNF of the first service is considered. In the second stage, the selection of the *backup server* for the first VNF of the first service is examined. In the $(2 \times U_1 + 1)^{\text{th}}$ stage, the selection of the main server for the first VNF of the second service is considered. We notice that U_1 is the number of VNFs in the SFC of the first service. In the final stage, the selection of the backup server for the last VNF of the last service is examined. In each stage, several candidates for the selection of the main or backup server for the respective VNF is determined. In the final stage, the certain selection of the main and backup server for all VNFs of all services is specified.

The state set of each stage, Z_n, is defined as a set of the servers where respective VNF can run on them. However, for our problem, this definition is incomplete in the even stages in which backup server selection is considered. In these stages, the NO may consider no backup server assignment as an option for the respective VNF. As a result, the set of states in the n^{th} stage is

$$Z_n = \begin{cases} \{0\} \bigcup S, & (n \bmod 2) = 0, \\ S, & (n \bmod 2) = 1 \end{cases}, n = 1, \ldots, L_V, \tag{34}$$

where S is the set of servers which introduced in (1) and $\{0\}$ indicates no server assignment which is usable in even stages.

One of the important challenges in using the VRSP algorithm is to define transition costs between all pairs of the states in the consecutive stages. The basis of transition cost for our problem is the objective function in (6). Let $\Theta_{n-1,n}^{i,j}$ denote the transition cost between the i^{th} state of the $(n-1)^{\text{th}}$ stage and j^{th} state of n^{th} stage. We can write $\Theta_{n-1,n}^{i,j}$ as

$$\Theta_{n-1,n}^{i,j} = r_k^u \times C_l + \psi_{n-1,n}^{i,j} + \phi_{n-1}^{i}, \tag{35}$$

where r_k^u indicates the required resource for the u^{th} VNF of the k^{th} service and C_l indicates the resource cost for the l^{th} InP. Also, k indicates the index of the considered service in the n^{th} stage, u indicates the index of the considered VNF of

TABLE I

NOTATION TABLE INCLUDING ALL NOTATIONS OF OPTIMIZATION PROBLEMS AND PROPOSED ALGORITHM

Symbol	Description
S, N	Set of server and links
$\|P\|$	Number of InPs
S_i^n, R_i^n	The n^{th} server of i^{th} InP and computational resource of S_i^n
$L_{i,j}^{n,m}$	Link between n^{th} server of i^{th} InP and m^{th} server of j^{th} InP
$C_{i,j}^{n,m}, B_{i,j}^{n,m}$	Cost and bandwidth of $L_{i,j}^{n,m}$
$C_i, \|S_i\|, v_i$	Cost of using servers, number of the servers and failure probability of the servers in i^{th} InP
U_k, b_k, r_k^u	Number of VNFs, bandwidth requirement and resource requirement of the u^{th} VNF for the k^{th} service
$x_{n,u}^{i,k}$	Decision variable of placing the u^{th} VNF of the k^{th} SFC in the n^{th} server of the i^{th} InP
ξ_s, ξ_l, ξ_T	Server, link, and total placement cost
f_k^u, e_k^u	Failure probability of the u^{th} VNF of the k^{th} service in initial and modified optimization problem
q_k, c_k	Probability of being in running state for the k^{th} service in initial and modified optimization problem
f_k	Failure probability of the k^{th} service
F_k, Q_k	Required failure probability and Required probability of being in running state for the k^{th} service
$y_{i,j,k}^{n,m,u}$	Binary variable for eliminating the non-linearity in (13)
L_V, Z_n	Number of stages and state set in the n^{th} stage
$\Theta_{n-1,n}^{i,j}, \psi_{n-1,n}^{i,j}$	Transition cost and link cost between the i^{th} state of the $(n-1)^{\text{th}}$ stage and j^{th} state of n^{th} stage
ϕ_n^j, τ_n^j	Cost and reliability for the j^{th} state of the n^{th} stage
k, u	Index of considered service and index of considered VNF in each stage
l, r	Index of considered InP and index considered server in respective InP for each state of each stage
$T_{n-1,n}^{i,j}$	Reliability when we are in the j^{th} state of the n^{th} stage, if we consider the transition between the i^{th} state of the $(n-1)^{\text{th}}$ stage and the j^{th} state of the n^{th} stage.
$D_{n-1,n}^{i,j}$	Decision metric for the path between the i^{th} state of the $(n-1)^{\text{th}}$ stage and j^{th} state of the n^{th} stage
Λ_n^j	Survived path in the j^{th} state of the n^{th} stage
I_n^j, XP_n^j	Index of the survived path and set of possible input states for the j^{th} state of n^{th} stage
OR_n^j	Objective reliability for the required reliability of the considered service in the j^{th} state of the n^{th} stage
M_k	Cost of exceeding the reliability requirement of k^{th} service
$\mathcal{P}$	Viterbi path which includes the placement characteristics

the k^{th} service in the n^{th} stage, and l indicates the InP index for the j^{th} state of the n^{th} stage. It is worth noting that for the odd values of n, we consider the main server selection for the u^{th} VNF of the k^{th} service and for the even values of n, we consider the backup server selection for the u^{th} VNF of the k^{th} service. Also, ϕ_{n-1}^i indicates the cost of being in the i^{th} state of the $(n-1)^{\text{th}}$ stage and $\psi_{n-1,n}^{i,j}$ indicates the link cost for the transition between the i^{th} state of the $(n-1)^{\text{th}}$ stage and the j^{th} state of the n^{th} stage. This cost can be written as

$$\psi_{n-1,n}^{i,j} = \tag{36}$$
$$\begin{cases} 0 & u = 1 \text{ or } j = 0 \\ b_k \times \left(C_{\Lambda_{n-1}^i[1,n-2],l}^{\Lambda_{n-1}^i[2,n-2],r} + C_{\Lambda_{n-1}^i[1,n-1],l}^{\Lambda_{n-1}^i[2,n-1],r} \right) & u \geq 2, n \in \text{odds} \\ b_k \times \left(C_{\Lambda_{n-1}^i[1,n-2],l}^{\Lambda_{n-1}^i[2,n-2],r} + C_{\Lambda_{n-1}^i[1,n-3],l}^{\Lambda_{n-1}^i[2,n-3],r} \right) & u \geq 2, n \in \text{evens} \end{cases},$$

where Λ_{n-1}^i is a $2 \times (n-1)$ matrix which indicates the main and backup server indices of the survived path in the i^{th} state of the $(n-1)^{\text{th}}$ stage. The first row of this matrix indicates the index of InP and the second row indicates the index of server. Also, r indicates the index of server in the j^{th} state of the n^{th} stage. When $j = 0$, we assume that $l = r = 0$ and we have $C_0 = 0$, $v_0 = 1$. Also, we assume that $C_{0,l}^{0,r} = 0$ for the case of no backup assignment in the survived path.

We define $T_{n-1,n}^{i,j}$ as the reliability level when we are in the j^{th} state of the n^{th} stage, if the VRSP algorithm moves from the i^{th} state of the $(n-1)^{\text{th}}$ stage to the j^{th} state of n^{th} stage. The value of $T_{n-1,n}^{i,j}$ is computed as

$$T_{n-1,n}^{i,j} = \tag{37}$$
$$\begin{cases} 1 - v_l & u = 1, n \in \text{odds} \\ 1 - \left(v_{\Lambda_{n-1}^i[1,n-1]} \times v_l\right) & u = 1, n \in \text{evens} \\ \tau_{n-1}^i \times (1 - v_l) & u \geq 2, n \in \text{odds} \\ \dfrac{\tau_{n-1}^i \times \left(1 - \left(v_{\Lambda_{n-1}^i[1,n-1]} \times v_l\right)\right)}{\left(1 - v_{\Lambda_{n-1}^i[1,n-1]}\right)} & u \geq 2, n \in \text{evens} \end{cases},$$

where τ_{n-1}^i is the reliability when we are in the i^{th} state of the $(n-1)^{\text{th}}$ stage. It is worth noting that in the first and second case of (37), the influence of main and backup servers assignment for the first VNF of the k^{th} service on the reliability of service is considered, respectively. Also, in the third and fourth case of (37), the influence of main and backup server assignment for the u^{th} VNF of the k^{th} service on the reliability of service is considered, respectively.

Now, we consider the survived path selection for all states of each stage. As mentioned before, in the VRSP algorithm, the survived path is selected according to a transition cost. The transition cost for our problem is the summation of placement cost, including server and link cost, and the cost of violating the reliability requirement of the considered service. As a result, the survived path for the j^{th} state of the n^{th} stage is selected by finding the path with minimum cost as

$$I_n^j = \underset{i \in XP_n^j}{\operatorname{argmin}} \left(D_{n-1,n}^{i,j}\right), \tag{38}$$
$$D_{n-1,n}^{i,j} = \Theta_{n-1,n}^{i,j} + M_k \times \left(OR_n^j - T_{n-1,n}^{i,j}\right) \times I\left(OR_n^j - T_{n-1,n}^{i,j}\right),$$

where $D_{n-1,n}^{i,j}$ is the total decision metric for the path between the i^{th} state of the $(n-1)^{\text{th}}$ stage and j^{th} state of the n^{th} stage, I_n^j indicates the index of the survived path and XP_n^j is the set of possible input states to j^{th} state of n^{th} stage. XP_n^j is the subset of the states set in the $(n-1)^{\text{th}}$ stage (i.e., $XP_n^j \subseteq Z_{n-1}$) which are having enough resources for running the considered VNF of the n^{th} stage in the respective server of the j^{th} state. Also, M_k is the cost of exceeding the reliability requirement of the k^{th} incoming service. It is worth noting that the value of M_k should be selected large enough compared to the cost of using servers and links to meet the required reliability. We will discuss the influence of this parameter on the performance of the VRSP algorithm in Section VI. The parameter OR_n^j is the objective reliability for the required reliability of the considered service in the j^{th} state of the n^{th} stage. In some states, we consider this parameter greater than the required reliability to guarantee the convergence of the algorithm to a placement in which the required reliability is met. The value of this parameter is defined as follows

$$OR_n^j = \begin{cases} Q_k & ; \quad j \neq 0, \\ 1 & ; \quad j = 0. \end{cases} \tag{39}$$

where $Q_k = 1 - F_k$ is the required reliability of the k^{th} service. According to (39), in $j = 0$, the path which leads to the maximum reliability $T_{n-1,n}^{i,j}$ is selected as a survived path. We notice that $j = 0$ occurs only in even stages where backup server selection is considered. We know that $j = 0$ means no backup server assignment for the considered VNF. We update the survived path in the j^{th} state of the n^{th} stage as

$$\Lambda_n^j[\,:, m] = \begin{cases} \Lambda_{n-1}^{I_n^j}[\,:, m] & 1 \leq m \leq n-1 \\ \{l, r\} & m = n \end{cases}, \tag{40}$$

where l and r indicate the indices of the considered InP and server in the j^{th} state of the n^{th} stage. Also, we can update ϕ_n^j and τ_n^j according to the index of survived path, I_n^j, as

$$\phi_n^j = \phi_{n-1}^{I_n^j} + \Theta_{n-1,n}^{I_n^j, j}, \tag{41}$$

$$\tau_n^j = \tag{42}$$

$$\begin{cases} 1 - v_l & u = 1, n = 2t-1 \\ 1 - \left(v_{\Lambda_{n-1}^{I_n^j}[1,n-1]} \times v_l\right) & u = 1, n = 2t \\ \tau_{n-1}^{I_n^j} \times (1 - v_l) & u \geq 2, n = 2t-1 \\ \dfrac{\tau_{n-1}^{I_n^j}\left(1 - \left(v_{\Lambda_{n-1}^{I_n^j}[1,n-1]} \times v_l\right)\right)}{\left(1 - v_{\Lambda_{n-1}^{I_n^j}[1,n-1]}\right)} & u \geq 2, n = 2t \end{cases}.$$

In the H^{th} stage where $H = \sum_{m=1}^{k} 2U_m$, the Viterbi path between the survived paths can be determined using

$$\mathcal{P} = \Lambda_H^{\zeta} \tag{43}$$

$$\zeta = \underset{i \in X_H}{\operatorname{argmin}} \Big(\phi_H^i + M_k \times \big(Q_k - \tau_H^i\big) \times I\big(Q_k - \tau_H^i\big)\Big)$$

where $\mathcal{P}$ is the Viterbi path which includes the placement characteristics for the first k services in the H^{th} stage, ζ is the index of Viterbi path and $I(.)$ is the indicator function which is 1 when its argument is greater than zero and zero otherwise. As mentioned before, the number of stages in each slot is L_V. The VRSP algorithm considers all of these stages step by step and after considering the $(L_V)^{\text{th}}$ stage, the Viterbi path for the placement of the services is selected. The number of candidates for the Viterbi path is $|Z_{L_V}|$, where $|Z_{L_V}|$ indicates the number of states in the last stage. In the last stage, the Viterbi path is determined using (43) with $H = L_V$.

Algorithm 1: Viterbi-based Reliable Service Placement (VRSP) Algorithm

```
InP input: ${|P|, S, L, |S_i|, C_i, v_i, R_i^n, C_{i,j}^{n,m}, B_{i,j}^{n,m}}$,
$i, j = 1, 2, \ldots, |P|, \quad n = 1, 2, \ldots, |S_i|, \quad m = 1, 2, \ldots, |S_j|$.
Service input:${K, U_k, b_k, r_k^u, F_k, M_k}$,
$k = 1, 2, \ldots, K, \quad u = 1, 2, \ldots, U_k$.
Viterbi algorithm input: $L_V = \sum_{k=1}^{K} 2U_k,\ Z_0 = 0,\ \phi_0^0 = 0$.
$ResourceIndicator = 1,\ H = L_V$.
$ServerRes_0^0[l, r] = R_l^r,\ l = 1, \ldots, |P|,\ r = 1, \ldots, |S_l|$.
for $(n = 1 : L_V)$ do
    Determine the index of considered service, $k$, and the index
      of considered VNF in the $k^{th}$ service, $u$.
    Determine the set of states, $Z_n$ using (34) .
    for $j \in Z_n$ do
        Determine the index of InP, $l$, and the index of server
          in the related InP, $r$, in the $j^{th}$ state of $n^{th}$ stage.
        $XP_n^j = Z_{n-1},\ RemovedState = \{\}$.
        if $j \neq 0$ then
            for $i \in XP_n^j$ do
                if $(ServerResource_{n-1}^i[l, r] - r_k^u < 0)$ then
                    $XP_n^j = XP_n^j - \{i\}$.
                end
            end
            if $b \bmod 2 = 0\ \&\&\ j \in XP_n^j$ then
                $XP_n^j = XP_n^j - \{j\}$.
            end
        end
        if $(XP_n^j \neq \phi)$ then
            for $i \in XP_n^j$ do
                Calculate $\Theta_{n-1,n}^{i,j}$ using (35) and (36).
                Calculate $T_{n-1,n}^{i,j}$ using (37).
            end
            Calculate $I_n^j$, using (38), and $\Lambda_n^j$, using (40).
            Calculate $\phi_n^j$, using (41), $\tau_n^j$, using (42).
            $ServerRes_n^j = ServerRes_{n-1}^{I_n^j}$.
            if $j \neq 0$ then
                $ServerRes_n^j[l, r] = ServerRes_n^j[l, r] - r_k^u$.
            end
        else
            Add the $j^{th}$ state to the $RemovedState$.
        end
    end
    if $(RemovedState = Z_n)$ then
        Set $H = \sum_{m=1}^{k-1} 2U_m$ and $ResourceIndicator = 0$.
        Determine the Viterbi path, $\mathcal{P}$, using (43) with $H$.
        break.
    else
        Remove all states in the $RemovedState$ from the $Z_n$.
    end
end
if $(ResourceIndicator = 1)$ then
    Determine the Viterbi path, $\mathcal{P}$, using (43), with $H$.
end
Determine the placement of service using Viterbi path $\mathcal{P}$.
```

In Fig. 2, an example of the VRSP algorithm is presented. There are two input services where each of them is a chain of two VNFs. Also, there are three InPs, each with one server. In this figure, we show the survived path for all states of the second stage with a red line. For example, the survived path for the third state of this stage is $I_2^3 = 1$. In the third stage, the total decision metrics of the first state for all possible input states, $D_{2,3}^{i,1}$, are indicated. The Viterbi path is indicated with a dashed red line. According to this Viterbi path, the main server of the first VNF of the first service is selected from 2th InP, and no backup server is considered for this VNF. Also, the main and backup servers for the second VNF of the first service are selected from 1th and 2th InPs, respectively. For the first VNF of the second service, the main and backup servers are selected from 3th and 2th InPs. Finally, the main server for the second VNF of the second service is selected from 1th InP, and no backup server is considered for this VNF.

Algorithm 1 shows the details of the proposed algorithm. This algorithm runs at the beginning of each slot and determines the placement of requested services in the corresponding slot. We assume that the requested services at the beginning of t^{th} slot, prolong for one slot. As a result, at the beginning of the $(t+1)^{\text{th}}$ slot, all resources of the servers and links are free, and the algorithm is run to determine the placement of

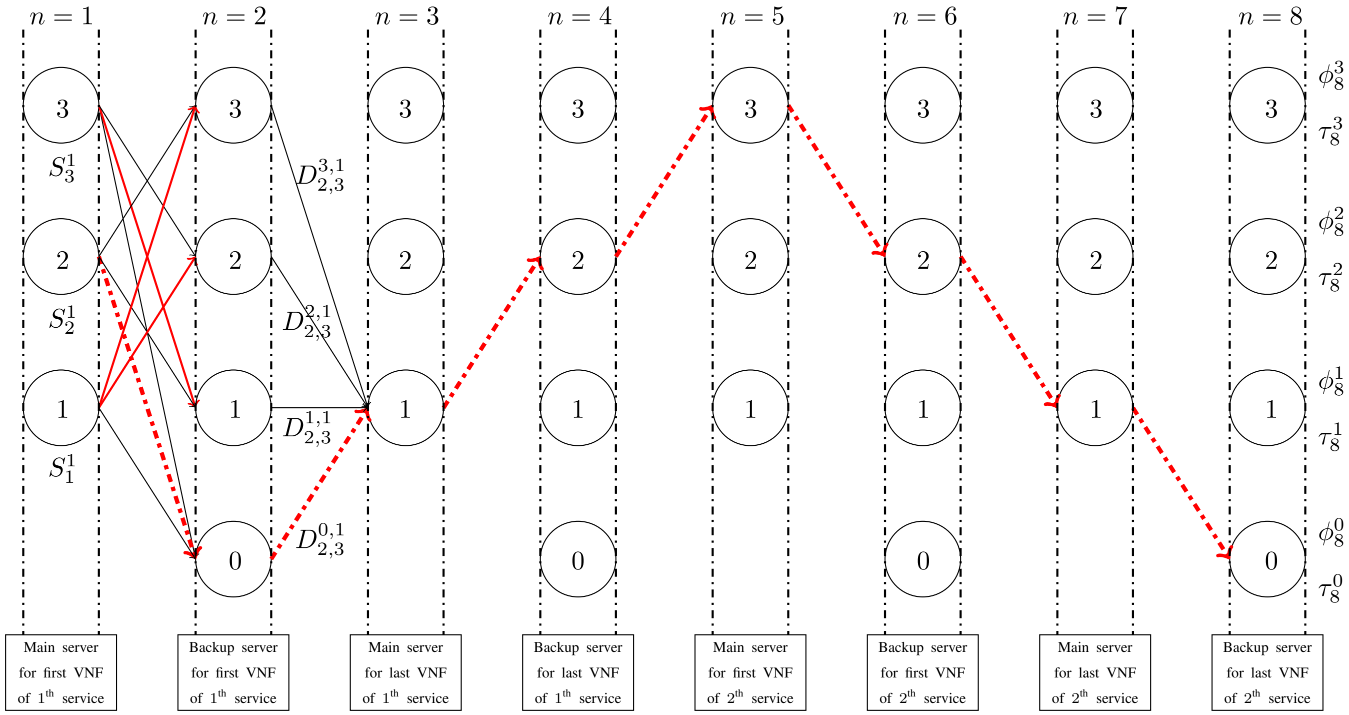

$n=1$
$n=2$
$n=3$
$n=4$
$n=5$
$n=6$
$n=7$
$n=8$
3
2
1
0
S_3^1
S_2^1
S_1^1
$D_{2,3}^{3,1}$
$D_{2,3}^{2,1}$
$D_{2,3}^{1,1}$
$D_{2,3}^{0,1}$
ϕ_8^3
τ_8^3
ϕ_8^2
τ_8^2
ϕ_8^1
τ_8^1
ϕ_8^0
τ_8^0
Main server for first VNF of 1th service
Backup server for first VNF of 1th service
Main server for last VNF of 1th service
Backup server for last VNF of 1th service
Main server for first VNF of 2th service
Backup server for first VNF of 2th service
Main server for last VNF of 2th service
Backup server for last VNF of 2th service

Fig. 2. An example of using VRSP for placement policy determination in a slot.

requested services at the beginning of $(t+1)^{\text{th}}$ slot.

At the beginning of the VRSP algorithm, we initialize the inputs and variables used in the algorithm. We consider three types of input. The first one is InP related inputs, which includes the capacity and cost of using servers and links. The second one is service related inputs, which includes the characteristics of the requested services. These characteristics include the number of services, the number of VNFs in each service, and the resource requirements of each VNF. Finally, we initialize the Viterbi related variables, which include the number of stages and a variable $ServerRes$, which determines the remaining resources of the servers in each stage. After initialization, we have a loop with a length of L_V. At each iteration of this loop, we consider the candidates for the main or backup server of a specific VNF. At the beginning of each iteration, in Line 9, we determine the index of considered service and the index of considered VNF in the corresponding service. Then, the set of states (i.e., candidates) for hosting the corresponding VNF is determined in Line 10.

In the following, we have an inner loop with the length $|Z_n|$ in Lines 11-38. At each iteration of the inner loop, we consider one of the candidates for selection of main or backup server for the corresponding VNF. At the beginning of the inner loop, in Line 12, the index of server and InP for the considered placement candidate, j is determined. Then, the set of possible input paths, XP_n^j is determined in Lines 14-23. For doing that, we should consider the reminder resource of servers of the input paths, $ServerRes_{n-1}^i$, in the $(n-1)^{\text{th}}$ stage. If the set of possible input paths to the j^{th} candidate is null, the corresponding server did not have enough resource for hosting the considered VNF and added to a set which is named $RemovedState$ in Line 36. Otherwise, between all possible input paths to the j^{th} candidate, the survived path, I_n^j, and the value of Λ_n^j, are determined in Line 29. Then, the value of Viterbi related parameter is updated in Lines 30-34. When all candidates for hosting the corresponding VNF is considered if the $RemovedState$ and Z_n are equal (Line 39), none of the servers have enough resource and as a result, the rest of services will not be admitted. This scenario can happen only in the odd stages in which the main server selection is considered. This is because in the even stages, we have no server assignment as an option, and as a result, the set of XP_n^j could not be null. In this scenario, the Viterbi path is determined between the survived paths of H^{th} stage (Lines 40-41). At the end of the outer loop, if all stages are considered, the Viterbi path is determined between the survived paths of the $(L_V)^{\text{th}}$ stage (Line 48). Finally, the placement of requested services is determined according to the Viterbi path using Algorithm 2, which is named Viterbi path placement (VPP).

Algorithm 2: Viterbi Path Placement (VPP)

```
Input: The Viterbi path $\mathcal{P}$, $H$, $S$,
$v_i$ for $i = 1, 2, \dots, |P|$. $K$, $U_k$ and $F_k$ for $k = 1, 2, \dots, K$.
$ServReliability = 1,\ AcceptServNum = 0.$
for $n = 1 : H$ do
    Determine the index of considered service, $m$, and the
     index of considered VNF in the $k^{\text{th}}$ service, $u$.
    if $(n = 2t - 1)$ then
        $l1 = \mathcal{P}[1, n]$, $r1 = \mathcal{P}[2, n]$.
    else
        $l2 = \mathcal{P}[1, n]$, $r2 = \mathcal{P}[2, n]$.
        $ServReliability = ServReliability \times (1 - v_{l1} v_{l2})$
        if $(u = U_m)$ then
            if $(ServReliability \geq (1 - F_m))$ then
                The $k^{\text{th}}$ service is admitted.
                $AcceptServNum = AcceptServNum + 1.$
            end
            $ServReliability = 1$
        end
    end
end
```

We calculate the reliability of the requested services using Algorithm 2. If the derived reliability is greater than the requirement, the service is accepted. Otherwise, the service is rejected. It is worth noting that we calculate the reliability for the services which are considered in the VRSP algorithm and have been assigned main and backup server. At the beginning of the VPP algorithm, we initialize the inputs and variables used in the algorithm. The most important input parameters are H and $\mathcal{P}$, which are the result of the VRSP algorithm. Then, we have a loop with a length of H. At the beginning of this loop, we determine the index of considered service and VNF in Line 5. Then, we determine the index of assigned InP and server as a main or backup server for the considered VNF (Lines 7 and 9). In the even iteration of the loop, we update the reliability of considered service (Line 10). If the considered VNF is the final VNF of service, we decide for accepting of considered service (Lines 11-17).

B. VRSP Algorithm for Desired Number of Backups

The proposed VRSP algorithm can be extended to a scenario in which the NO can consider the assignment of W backup servers to each VNF for meeting the reliability requirement of the incoming services. In this scenario, the number of the stages is $L_V = \sum_{k=1}^{K} (W+1)U_k$. In the first stage, the selection of the main server for the first VNF of the first service is considered. In the second stage, the selection of the first backup server for the first VNF of the first service is examined. In the third stage, the placement of the second *backup server* for the first VNF of the first service is examined. In the $(W \times U_1 + 1)^{\text{th}}$ stage, the placement of the main server for the first VNF of the second service is considered. Also, the set of states in the n^{th} stage is

$$Z_n = \begin{cases} \{0\} \bigcup S, & \big(n \bmod (W+1)\big) \neq 1, \\ S, & \big(n \bmod (W+1)\big) = 1 \end{cases}. \tag{44}$$

As observed in (44), each one of the VNFs can have from zero to W backups. The definitions of $\Theta_{n-1,n}^{i,j}$, $T_{n-1,n}^{i,j}$ and $D_{n-1,n}^{i,j}$ are the same as one backup scenario. However, some straightforward modifications should be applied for the computation of these parameters.

C. Complexity Analysis

In this part, we would like to investigate the computational complexity of the proposed VRSP algorithm. As mentioned before, the revised version of the initial optimization problem in (24)-(33), can not be solved using a polynomial time algorithm. As a result, we introduce the computational complexity of the Brute-force search for solving the initial optimization problem. Let $|P|$ indicate the number of InPs, $|S|$ be the maximum number of the servers in an InP $\big(i.e., |S| = \max_{i=1,2,\ldots,|P|}\{|S_i|\}\big)$, and K denotes the number of the incoming services in a slot and U be the number of VNFs in a service. Also, we consider only one backup server for each VNF. As a result, the number of binary decision variables is $2UK$. The computational complexity of the Brute-force search is $O\big((|P| \times |S|)^{2UK} \times 2UK\big)$, which grows exponentially with the increase in the number of incoming services and their VNFs. On the hand, the complexity of the proposed VRSP algorithm is $O\big(2UK \times (|P| \times |S|)^2\big)$ which grows linearly with the increase in the input size. For the case in which the NO considers the assignment of W backup servers to each VNF, the complexity of the proposed VRSP algorithm is $O\big((W+1)UK \times (|P| \times |S|)^2\big)$ which grows linearly with the increase in the maximum number of the backup servers for each VNF.

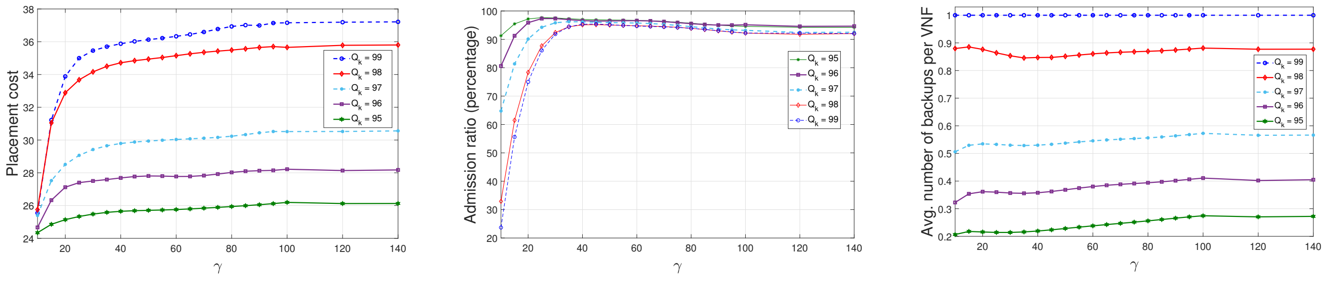
VI. Numerical Results

In this section, we evaluate the performance of VRSP algorithm using numerical experiments. We first consider the effect of the parameter M_k, in the performance of the proposed algorithm. Then, we compare the result of the proposed algorithm with the optimal solution. Finally, we evaluate the performance of the proposed algorithm in different scenarios for the different number of service requests and compare it with the existing reliability-aware service placement methods. We consider three metrics for evaluating the performance of the algorithms, which includes the placement cost introduced in (6), the number of backup servers, and the admission ratio. The number of backup servers is the number of additional servers used to meet the required reliability. The admission ratio is defined as the ratio of the number of accepted services with the required reliability to the number of incoming services. Furthermore, all the simulations whose results are reported in this paper are conducted using a machine having an Intel 2.7 GHz processor and 8 GB of RAM.

A. VRSP Parameters Evaluation

The parameter M_k is defined as the cost of violating the reliability requirement for the k^{th} incoming service. In this part, we evaluate the effect of this parameter on the performance of the VRSP algorithm using three introduced metrics. We define $M_k = \gamma \times C_1$, where C_1 is the cost of using the servers of the first InP, which has minimum reliability and γ is a positive integer value. The value of γ indicates the ratio of the cost of violating the required reliability and the cost of using the first InP. With an increase in the value of γ, the placement cost of services will be negligible compared to the cost of violating the required reliability which can lead to an increase in the admission ratio. We consider seven InPs with different reliability levels from 93% to 99% with steps of 1%. Each InP has three servers with identical reliability levels. For each server, we consider a capacity of 100 units. For the incoming services, we consider a Poisson process with mean 12 services per slot. The SFC of each service consists of three to six VNFs. Also, we assume that the required reliability of the services is among $\{95, 96, 97, 98, 99\}$, according to the SLA requirement of Google Apps [28]. The resource demand of each VNF is random between 10 and 30 units.

In Fig. 3(a), the placement cost, including main and backup servers and in Fig. 3(b), admission ratio for different values of γ are shown for simulation of 100,000 time slots. The execution time of the proposed VRSP algorithm to place the incoming services in each slot is 19 ms. As seen in Fig. 3(a), minimum placement cost is achieved for small values of γ. This is because, for small values of γ, the services with low resource requirement are accepted. In other words, for small

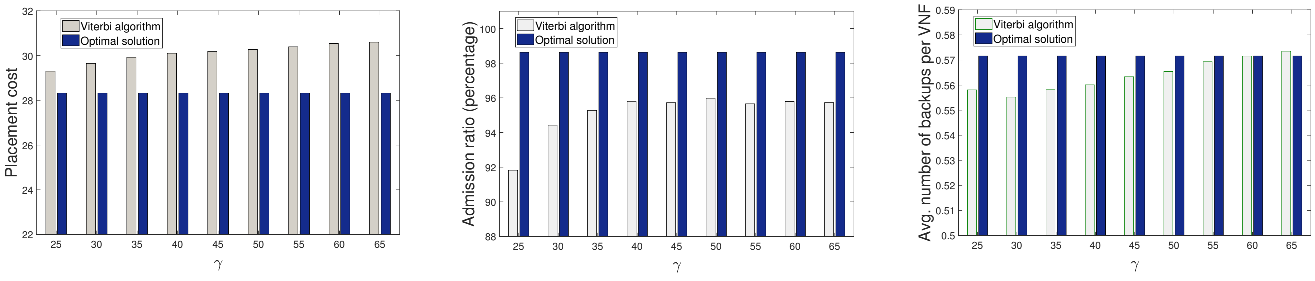


(a) Placement cost versus γ.

(b) Admission ratio versus γ.

(c) Number of backups per VNF versus γ.

Fig. 3. Performance of the VRSP for different values of γ



(a) Placement cost versus γ.

(b) Admission ratio versus γ.

(c) Number of backups per VNF versus γ.

Fig. 4. Performance comparison of the VRSP for different values of γ and Optimal solution.

values of γ, the cost of exceeding the required reliability of the service is not large enough compared to the cost of servers and links to force the algorithm for accepting all services, if feasible. As seen in Fig. 3(b), the admission ratio of highly reliable services (i.e., $0.98-0.99$) requiring more backup servers compared to other services is very low. With the increase in the value of γ, the number of admitted services and placement cost are increased. This increase is significant for highly reliable service requests (i.e., $0.98-0.99$). For large values of γ, because of the unnecessary backups, the number of accepted services is decreased, and the placement cost is increased as seen in Figs. 3(a) and 3(b).

In Fig. 3(c), the number of backups for different values of γ is shown. In the beginning, increasing the value of γ increases the number of backups because of the growth in the number of accepted services. In the following, increasing the value of γ decreases the number of backups because of the assignment of the appropriate server as a main or backup for the VNFs. For example, assigning a high reliable server as the main server to one VNF of service can obviate the need for assigning the backup server to other VNFs of the same service. For large values of γ, increasing the value of γ increases the number of backups because of the unnecessary backups.

B. VRSP Optimality Evaluation

In this part, we would like to compare the performance of the VRSP algorithm with the optimal solution. The considered infrastructure network consists of five InPs with different reliability from 95% to 99% with 1% step. Each one of the InPs has two servers with identical reliability. For each server, we consider a capacity of 100 units of one resource type. The incoming services are considered as a Poisson process with the mean of five services per slot. The SFC of each service consists of three to four VNFs. Also, the required reliability of the services is among $\{95, 96, 97, 98, 99\}$, according to the SLA requirement of Google Apps [28]. The resource demand of each VNF is random between 10 and 30 units. It is worth noting that because of the highly computational characteristic of solving MICP, the considered simulation setup is simple. For solving the introduced MICP, we use CVX [29] and Gurobi [7] which are used for solving convex programming.

In Fig. 4(a), the placement cost, including main and backup servers and in Fig. 4(b), the admission ratio for different values of γ are shown for the VRSP algorithm and the optimal solution. For small values of γ, the services with low resource requirement are accepted. As a result, the admission ratio of VRSP is far from the optimal solution, and the placement cost of VRSP is close to the optimal solution, as seen in Fig. 4(b) and Fig. 4(a), respectively. It is worth noting that for small values of γ, there is enough server resource and bandwidth in the InPs for accepting most of the services. But in the VRSP algorithm, the server resource and bandwidth cost of some services is higher than the cost of rejecting services, and are hence rejected. With the increase in the value of γ, services with high resource requirement are also accepted, and the admission ratio and placement cost are increased. With more increase in the value of γ, because of the unnecessary backups, the admission ratio is decreased, and the placement cost is increased.

In Fig. 4(c), the mean number of backup servers for different values of γ is shown for VRSP and optimal solution. In small values of γ, the number of backup servers of the VRSP algorithm is less than the optimal solution because of rejecting most of the high-reliability requirement services and using the highly reliable server for placement of incoming service. However, with the increase in the value of γ, the admission ratio, especially for high-reliability requirement services and as a result, the number of backups increased. Even though for most values of γ, the number of backup for VRSP is lower than the optimal solution and the placement cost of the VRSP algorithm is higher than the optimal solution because of using the highly reliable server in VRSP algorithm.

From the perspective of maximizing the admission ratio, the best performance of the VRSP algorithm is achieved using $\gamma = 50$. For this value of γ, the shortages of admission ratio and placement cost in the VRSP algorithm compared to the optimal solution are 2.69% and 6.86%, respectively. The average number of backup servers in the VRSP algorithm is lower than the optimal solution by 1.1%.

C. VRSP Algorithm versus Existing Methods

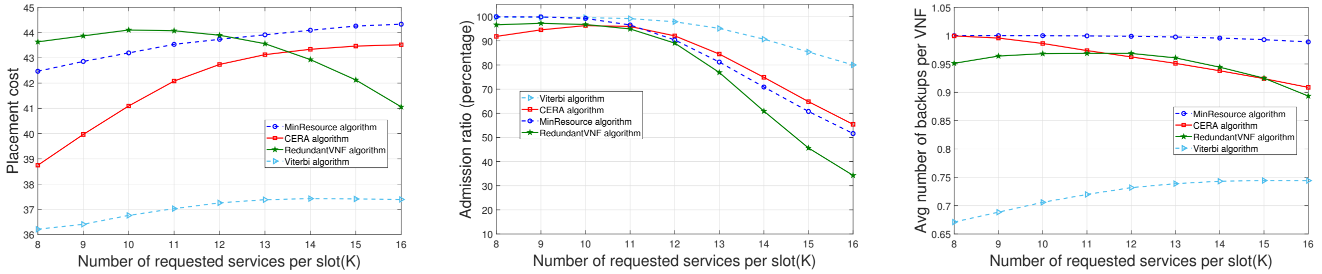
In this part, we compare the performance of VRSP with existing methods of reliability-aware service placement in NFV. Similar to the previous parts, the performance metrics are placement cost, admission ratio, and the number of backup servers. We compare the performance of the proposed algorithm with three heuristic algorithms. The first one is named MinResource, which selects the VNF with the minimum resource and then chooses a backup server for the given VNF in a way that the reliability requirement is met. However, if it is infeasible, the selected VNF will be backed up at the server with the highest reliability and sufficient resources [23]. The second heuristic algorithm is named CERA introduced in [23], which uses cost importance measure for VNF selection for backup and backup placement in InP at each iteration until the required reliability is met. The third heuristic algorithm is named RedundantVNF introduced in [25], considers iterative backup adding to the services according to the reliability of each VNF in each service.

The number of incoming services has a significant impact on the service placement. In the following, we evaluate the performance of the proposed algorithm for a different number of incoming services. For the infrastructure network, we consider seven InPs with different reliability from 93% to 99% with 1% step, respectively. Each of the InPs has five servers with identical reliability. For each server, we consider a capacity of 100 units of one resource type. The SFC of each service consists of three to six VNFs. Also, the required reliability of the services is among $\{95, 96, 97, 98, 99\}$, according to the SLA requirement of Google Apps [28]. The resource demand of each VNF is random between 20 and 40 units.

In Figs. 5(a)-5(b), the placement cost and the admission ratio for the different number of requested services are indicated for VRSP and three existing methods. The results of the VRSP algorithm are achieved by setting $\gamma = 45$ which is an appropriate value for optimizing the three introduced metrics. As observed in Fig. 5(a), the average improvement in the placement cost achieved by the VRSP algorithm compared to the MinResource, RedundantVNF, and CERA algorithms is 15.1%, 14.3%, and 11.7%, respectively. Also, as observed in Fig. 5(b), the average improvement in the admission ratio achieved against the MinResource, RedundantVNF, and CERA algorithms is 12.3%, 19.7%, and 12.1%, respectively. When the number of requested services is low, almost all incoming services are accepted by using VRSP and the other three methods, as shown in Fig. 5(b) and the placement cost of the VRSP algorithm is remarkably lower than the other three existing methods, as shown in Fig. 5(a). Also, the placement cost of CERA is lower than the other two methods, as indicated in Fig. 5(a). As seen in Fig. 5(b), with the increase in the number of requested services, the number of accepted services for three existing methods is severely decreased. This decrease in VRSP is remarkably lower than the existing methods. In other words, when the number of incoming services is increased, the superiority of VRSP compared to the existing methods emerges. Also, with the increase in the number of the requested services, the placement cost of the VRSP, MinResource, and CERA algorithms is increased in a way that the superiority of VRSP is enhanced, as seen in Fig. 5(a). This increase in the placement cost of the CERA is higher than the MinResource algorithm. On the other hand, the placement cost of the RedundantVNF algorithm is decreased by increasing the number of requested services due to the severe reduction in the admission ratio of this algorithm. It is worth noting that by increasing the number of requested services, the RedundantVNF algorithm only admits services with a low number of VNFs and resource requirement.

In Fig. 5(c), the mean number of backup servers for the different number of requested services is shown. As observed in this figure, the average improvement in the number of backup servers achieved by the VRSP algorithm compared to the MinResource, RedundantVNF, and CERA algorithms is 27.7%, 24%, and 24.7%, respectively. The number of backup servers for VRSP increases with the number of requested services, but this converges to 0.75. As observed in Fig. 5(c), the number of backup servers decreases with an increasing number of requested services for the RedundantVNF and CERA methods. With an increasing number of requested services, the probability of admitting services that need a lower number of backup servers to meet their reliability requirement is increased. This leads to a reduction in the number of backup servers for these two methods.

The number of VNFs in the incoming services has a significant impact on the placement. In the following, we evaluate the performance of the proposed algorithm against the number of VNFs in the service. The simulation setup is the same as the introduced setup at the beginning of Section VI-C, except the following. For the incoming services, we consider a Poisson process with mean 12 services per slot. The SFC of each service consists of two to six VNFs. In Figs. 6(a)-6(c), the placement cost, the admission ratio, and the mean number of backup servers for different number of VNFs in the incoming services are shown for VRSP and three existing methods, respectively. The results of the VRSP algorithm are achieved by selecting $\gamma = 50$, which is an appropriate value for optimizing three introduced metrics. It should be mentioned that in the experiments of Figs. 5(a)-5(c), we chose a value for γ, which can lead to the best average performance over the different number of the service requests. However, the best performance of the VRSP algorithm, when the number of services requests is 12 can be achieved by using $\gamma = 50$ in Figs. 6(a)-6(c). It is worth noting that the performance deviation of the VRSP algorithm by varying γ

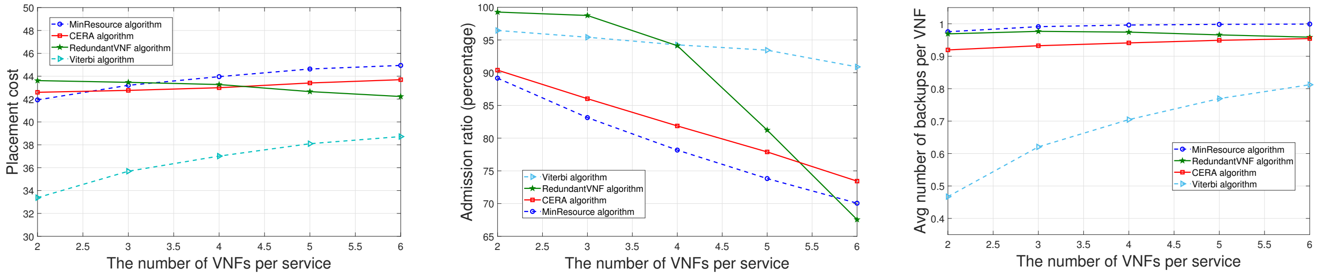


(a) Placement cost versus number of incoming services.

(b) Admission ratio versus number of incoming services.

(c) Number of backups per VNF versus number of incoming services.

Fig. 5. Performance comparison of the VRSP algorithm and three other methods for various number of the incoming services.



(a) Placement cost versus number of VNFs in the incoming services.

(b) Admission ratio versus number of VNFs in the incoming services.

(c) Number of backups per VNF versus number of VNFs in the incoming services.

Fig. 6. Performance comparison of the VRSP algorithm and three other methods for various number of the VNFs in incoming services.

between 40 and 50 is negligible.

As observed in Fig. 6(a), the average improvements in the placement cost by increasing the number of VNFs, using VRSP algorithm compared to the MinResource, RedundantVNF, and CERA algorithms are 16.4%, 15%, and 15.1%, respectively. Also, according to Fig. 6(b), the average improvements in the admission ratio by increasing the number of VNFs, using VRSP algorithm compared to the MinResource, RedundantVNF, and CERA algorithms are 16.3%, 6.5%, and 13%, respectively. Finally, as observed in Fig. 6(c), the average improvement in the number of backup servers achieved against the three alternative approaches is 32.1%, 30.3%, and 28.3%, respectively. According to Figs. 6(a) and 6(c), with the increase in the number of the VNFs per service, the improvement in the placement cost and the mean number of the backup servers for the proposed algorithm against the three existing methods is slightly decreased. This is due to the increasing number of required backups, which leads to an increased placement cost. The main reason for this increase is that the reliability of the InPs is in the proximity of the reliability requirement of the services. With the increase in the value of the reliability requirement and the number of the VNFs in the incoming services, the VRSP algorithm has to allocate the backup server to most or all of the VNFs of the service. On the other hand, according to Fig. 6(b), for the low number of VNFs per service, the admission ratio of the RedundantVNF algorithm is slightly greater than the proposed algorithm. However, with the increase in the number of VNFs, the admission ratio of the proposed VRSP algorithm is remarkably greater than the three existing methods, especially the RedundantVNF algorithm.

VII. Conclusion

This paper establishes a new reliability-aware service placement for NFV-enabled NOs in a multi-InP scenario. We proposed an optimization problem for simultaneously selecting main and backup servers of a service chain, considering the cost of using InP resources and constraints on the reliability requirement of the services. We reformulated the initial problem as an MICP, which is more tractable. Then, we proposed a sub-optimal algorithm named VRSP with less complexity. Performance evaluations show the proximity of the VRSP algorithm to the optimal solution. For comparison, we used three metrics: the placement cost, the admission ratio, and the number of backup servers. The evaluation results show that the proposed VRSP algorithm is robust to an increasing number of service requests and number of the VNFs in the SFCs. More precisely, the improvement of the admission ratio and placement cost compared to the existing methods is mainly achieved with an increased number of service requests. As the future work, we are planning to consider the scenarios in which the departure time of the services is random, and the NO does not have complete information about the reliability of the InPs. In such scenarios, the NO gradually learns about the departure time of the services and reliability of the InPs and improves the service admission and placement policy.

References

[1] J. G. Herrera and J. F. Botero, “Resource allocation in NFV: A comprehensive survey,” *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[2] J. Sherry, S. Ratnasamy, and J. S. At, “A survey of enterprise middlebox deployments,” Technical Report UCB/EECS-2012-24, EECS Department, University of California, Berkeley, 2012.

[3] G. NFV, “Network functions virtualisation (NFV); architectural framework,” *NFV ISG*, Oct. 2013.

[4] 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 Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2016.

[5] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici, "Clickos and the art of network function virtualization," in *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, San Diego, CA, Aug. 2014.

[6] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. of IEEE 3rd CloudNet*, Luxembourg, Luxembourg, Oct. 2014.

[7] I. Gurobi Optimization, "Gurobi optimizer reference manual; 2015," *URL: http://www. gurobi. com*, 2016.

[8] I. C. O. Studio, "Software, 2012," *URL: http://www-01. ibm. com/software/integration/optimization/cplex-optimization-studio*.

[9] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach," *IEEE Trans. on Services Computing*, 2017.

[10] M. Mechtri, C. Ghribi, and D. Zeghlache, "A scalable algorithm for the placement of service function chains," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 533–546, 2016.

[11] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Trans. on Network and Service Management*, vol. 13, no. 4, pp. 725–739, 2016.

[12] V. Eramo, M. Ammar, and F. G. Lavacca, "Migration energy aware reconfigurations of virtual network function instances in NFV architectures," *IEEE Access*, vol. 5, pp. 4927–4938, 2017.

[13] S. Gu, Z. Li, C. Wu, and C. Huang, "An efficient auction mechanism for service chains in the NFV market," in *Proc. of IEEE INFOCOM*, San Francisco, CA, Apr. 2016.

[14] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. C. Lau, "Online stochastic buy-sell mechanism for vnf chains in the NFV market." *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 392–406, 2017.

[15] S. DOro, L. Galluccio, S. Palazzo, and G. Schembra, "Exploiting congestion games to achieve distributed service chaining in NFV networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 407–420, 2017.

[16] A. Leivadeas, G. Kesidis, M. Falkner, and I. Lambadaris, "A graph partitioning game theoretical approach for the vnf service chaining problem," *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 890–903, 2017.

[17] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proc. of IEEE INFOCOM*, Hong Cong, Hong Cong, Apr. 2015.

[18] D. Dietrich, A. Abujoda, and P. Papadimitriou, "Network service embedding across multiple providers with nestor," in *Proc. of IEEE IFIP Networking*, Sophia Antipolis, France, Jul. 2015.

[19] D. Dietrich, C. Papagianni, P. Papadimitriou, and J. S. Baras, "Network function placement on virtualized cellular cores," Bangalore, India, Jan. 2017.

[20] R. Riggio, A. Bradai, D. Harutyunyan, T. Rasheed, and T. Ahmed, "Scheduling wireless virtual networks functions." *IEEE Trans. Network and Service Management*, vol. 13, no. 2, pp. 240–252, 2016.

[21] S. Herker, X. An, W. Kiess, S. Beker, and A. Kirstaedter, "Data-center architecture impacts on virtualized network functions service chain embedding with high availability requirements," in *Workshop. of IEEE Globecom*, San Diego, CA, Dec. 2015.

[22] J. Fan, Z. Ye, C. Guan, X. Gao, K. Ren, and C. Qiao, "Grep: Guaranteeing reliability with enhanced protection in NFV," in *Proc. of ACM SIGCOMM Workshop*, Heraklion, Greece, July. 2015.

[23] W. Ding, H. Yu, and S. Luo, "Enhancing the reliability of services in NFV with the cost-efficient redundancy scheme," in *Proc. of IEEE ICC*, Paris, France, May. 2017.

[24] Y. Kanizo, O. Rottenstreich, I. Segall, and J. Yallouz, "Optimizing virtual backup allocation for middleboxes," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2759–2772, 2017.

[25] L. Qu, C. Assi, K. Shaban, and M. J. Khabbaz, "A reliability-aware network service chain provisioning with delay guarantees in NFV-enabled enterprise datacenter networks," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 554–568, 2017.

[26] J. Fan, M. Jiang, and C. Qiao, "Carrier-grade availability-aware mapping of service function chains with on-site backups," in *Proc. of IEEE/ACM IWQoS*, Vilanova i la Geltru, Spain, June. 2017.

[27] H. R. Khezri, P. A. Moghadam, M. K. Farshbafan, V. Shah-Mansouri, H. Kebriaei, and D. Niyato, "Deep q-learning for dynamic reliability aware nfv-based service provisioning," *arXiv preprint arXiv:1812.00737*, 2018.

[28] "Google apps service level agreement," [Online]. Available:http://www.google.com/apps/intl/en/terms/sla.html.

[29] M. Grant, S. Boyd, and Y. Ye, "Cvx: Matlab software for disciplined convex programming (2008)," *Web page and software available at http://stanford. edu/ boyd/cvx*, 2015.

Mohammad Karimzadeh Farshbafan received the B.Sc. degree in electrical engineering from Amirkabir University of Technology, Tehran, Iran in 2012 and M.Sc. degree in electrical engineering from Sharif University of Technology, Tehran, Iran in 2014. Since September 2015, he is a PhD candidate in electrical engineering in the University of Tehran. His main concern during Ph.D. program is on softwarization techniques including software-defined networking and network function virtualization for the next generation of telecommunication networks.

Vahid Shah-Mansouri (S02M13) received the B.Sc. degree in electrical engineering from the University of Tehran, Tehran, Iran, in 2003, the M.Sc. degree in electrical engineering from Sharif University of Technology, Tehran, Iran, in 2005, and the Ph.D. degree from The University of British Columbia, Vancouver, BC, Canada, in 2011, respectively. Since 2013, he has been an Assistant Professor with the School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran. His research interests include analysis and mathematical modeling of communication and computer networks.

Dusit Niyato (M'09-SM'15-F'17) is currently a professor in the School of Computer Science and Engineering, at Nanyang Technological University, Singapore. He received B.Eng. from King Mongkuts Institute of Technology Ladkrabang (KMITL), Thailand in 1999 and Ph.D. in Electrical and Computer Engineering from the University of Manitoba, Canada in 2008. His research interests are in the area of energy harvesting for wireless communication, Internet of Things (IoT) and sensor networks.