Received 19 September 2022, accepted 28 September 2022, date of publication 4 October 2022, date of current version 12 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3211983

RESEARCH ARTICLE

SFC-HO: Reliable Layered Service Function Chaining

XUANZHE CHEN, JINHE ZHOU, AND SONGLIN WEI
Key Laboratory of the Ministry of Education for Optoelectronic Measurement Technology and Instrument, Beijing Information Science & Technology University, Beijing 100101, China
School of Information and Communication Engineering, Beijing Information Science & Technology University, Beijing 100101, China

Corresponding author: Jinhe Zhou (zhoujinhe@bistu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61872044 and Grant 61901043.

ABSTRACT Network function virtualization (NFV) utilizes IT virtualization technology to realize the functions of various network devices and realize the decoupling of hardware and software functions. Typically, a service request specifies the virtual network functions (VNFs) it needs and the order between them. A network flow needs to traverse a set of sequential VNFs called a service function chain (SFC). Although SFC can increase the flexibility of service orchestration to meet the needs of different users, network providers face many challenges due to the need to ensure service reliability and some constraints. Therefore, how orchestrating SFC and designing a suitable network architecture is a critical issue in this field at present. We propose an efficient network layer-based SFC orchestration method, called SFC-hierarchical orchestration (SFC-HO). Our method separates virtual and physical networks into layers and computes the availability of resources in each layer. We filter the VNFs of each layer and deploy them to the physical layer that meets the greatest benefit. To this end, we formulate the problem as an integer linear programming (ILP) problem to minimize the total deployment cost. In order to further optimize the layering strategy, we innovatively use the Benders decomposition method to decompose the SFC-HO problem into two sub-problems, which are the hierarchical mapping problem of VNFs and the routing problem between nodes. We propose two algorithms for the two problems respectively. The simulation results show that compared with the SFC orchestration process in the traditional network model. Our research can effectively improve the reliability of the SFC, reduce the delay of the routing process, and effectively reduce the cost consumption. Our research effectively solves the problem of difficult service orchestration for operators in the face of diverse service demands and a large number of service requests.

INDEX TERMS Network function virtualization, service function chaining, software-defined network, layered orchestration.

I. INTRODUCTION

The high development of network technology in the past decade has not only made the network architecture more complex but also caused a sharp increase in network traffic. It is difficult for hardware networks to sustain the rapidly increasing traffic and diversified services. In traditional network architectures, the provision of network functions is more dependent on dedicated physical devices, which leads to problems of high coupling between devices and low flexibility of the devices. Traditional networks that rely on dedicated equipment are difficult to provide diversified services. Therefore, scalability, flexibility, and energy consumption are urgent issues to be considered in the deployment of network functions. For this reason, it is imperative to convert the traditional physical equipment into software to solve many problems existing in the physical equipment. Network function virtualization (NFV) uses general-purpose hardware such as x86 and virtualization technologies to carry software processing of many functions [1]. Thereby reducing the cost of expensive network equipment. The concept of NFV was originally proposed by the European telecommunications

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka.

standards institute (ETSI) to reduce costs and accelerate service deployment for network operators [1]. NFV transforms traditional networking by using virtualization and cloud technologies to separate network functions from hardware and abstract network services into software called virtualized network functions (VNFs) that run on basic hardware [2].

VNF is a software-based network element instantiated in a virtual machine (VM) or a container running on a commodity server through standard IT virtualization techniques [3]. Different from traditional network functions, it is a network element built on a virtual machine and does not depend on proprietary hardware devices. Multiple virtual machines can be built on one hardware device, and each VM can implement at least one network function. Deploying virtualized network functions in this way can decouple network functions from proprietary hardware, reduce the number of physical devices, reduce energy consumption, and improve network scalability. Therefore, using VNFs generated by NFV technology to replace traditional network functions that rely on dedicated physical devices is an important means to solve the above-mentioned existing problems.

Network services usually consist of several functions through which traffic flows in a specific order. In traditional network architectures, service providers select dedicated physical devices based on the needs of each function. In general, each function corresponds to a physical device, which may come from different vendors [4]. The service requested by the user consists of these abstract functions and the virtual links connecting them. In this paradigm, VNFs are connected according to a certain functional sequence and connected with source nodes and destination nodes to provide complete services, forming a service function chain (SFC). SFC enables service providers to dynamically configure and change deployed network services without changing physical wiring [5]. Therefore, the service provided by SFC is more efficient.

Usually, customers require services that are highly reliable and fast. Although SFC can provide diversified services, how to make the service more reliable has always been the focus of current research. We consider the availability of VNFs and the reliability of SFCs. Generally speaking, reliability includes software availability and infrastructure reliability. In the case of sufficient hardware resources, the higher the availability of the set of VNFs that make up the SFC, the higher the reliability of the SFC. When we consider the question of reliability, we must consider the reliability of virtual resources and the reliability of physical resources. Designing an SFC architecture that is more in line with resource scheduling in the SFC construction phase is a common method to solve reliability problems. In addition, the backup of network resources is also widely used to improve the reliability of SFC.

Although SFC makes it easier to provide customized services, it also leads to end-to-end vulnerabilities. In the process of orchestrating SFC, the failure of any VNF instance will lead to the failure of the entire chain, which greatly affects the efficiency of service deployment. When a VNF fails, traffic needs to be rerouted to other function instances, which will result in higher latency and energy consumption. SFC orchestration needs to consider issues such as VNF composition and screening, network bandwidth, and computing resources. In the process of VNF scheduling, we should give preference to VNFs with higher availability. We consider that the deployment of VNF requires computing resources and bandwidth resources. Because the traffic of the service function should not consume too many network resources and should be able to occupy fewer resources on the host and virtual machines. In addition, we consider that a service must meet the requirements of low latency as much as possible under the premise of reliable service. In the process of providing services, the first task is to ensure that the services can be fully provided to customers. At this point, it is necessary to ensure the availability of various functions. Secondly, it is necessary to ensure the lowest possible delay. The high-latency SFC violates the original intention of providing services under the virtualized network architecture. From the perspective of the operator's revenue, when using SFC to route traffic to provide services, it is also necessary to ensure that the delay and cost are as low as possible under the premise of service availability. Therefore, in our paper, we first consider the availability of SFC and consider the optimization problems of reliability and cost while satisfying constraints such as delay and bandwidth. Research shows that the SFC deployment problem is an NP-hard problem [6]. Generally, an approximate solution is obtained using efficient heuristics.

Currently, there are some studies on the construction of SFC and network model construction. However, in terms of availability and reliability the existing research has the following shortcomings:

- Regarding the construction of SFC, the existing research mostly considers the scenario of a single topology, and the network model is not optimized. These models lack consideration for flexible orchestration of SFC. Furthermore, the current network model is difficult to deal with the mapping problem between a large number of virtual nodes and mixed physical nodes. Lack of structural division between physical devices.
- Regarding SFC backup, existing researches focus on increasing the availability of VNF instances by adding backups, but lacks cost considerations, resulting in low resource utilization.

Aiming at the shortcomings of existing research, this paper comprehensively solves the problems of SFC reliability and cost by researching the layered model of SFC. We propose a layered model of the network to more rationally place virtual and physical resources in the network during the mapping phase of the VNF. We propose a hierarchical SFC orchestration method, which can effectively improve the reliability of orchestration while saving backup resources. Our algorithm transforms the network topology into a multi-layer topology to make VNFs deployment more convenient.

The main contributions of this paper are as follows:

- We define the SFC-HO problem and propose a rich ILP model for it.
- We analyze the proposed model to hierarchically partition virtual and physical networks, and build SFC as a hierarchical directed graph. We construct our mathematical model with parameters such as node availability, latency, and cost, and formulate the objective function for VNF mapping.
- We use the Benders decomposition method to decompose the SFC-HO problem into two sub-problems, which are the hierarchical mapping problem of VNFs and the routing problem between nodes.
- Using extensive simulations, we compare the reliability-aware hierarchical routing (RAHR) algorithm to exact heuristics. The results show that our network model helps to orchestrate SFC more flexibly. In addition, our algorithm can improve the reliability of SFC to the greatest extent under the premise of lower cost.

The rest of this article is organized as follows. We review related work in section II. In section III, the related network models and formulations are defined. In section IV, a detailed description of our proposed VNF mapping method is given, and we also explain the SFC-HO method in detail. In section V, we describe our algorithm in detail and formulate the RAHR algorithm. The simulation results are shown in section VI. We conclude this article in section VII.

II. RELATED WORKS

In this section, we analyze the related work in the past. We contrast some of the most contrasting work with the issues considered in this article. Table 1 shows the aspects covered by the research content of the references.

A. SFC SCHEDULING

A lot of research has been done on VNF placement and SFC orchestration by many researchers. When considering the resource scheduling problem in SFC, many researchers consider the method of sharing and backup resources. In the study of SFC scheduling, many resource backup schemes have been proposed and reliability has been considered. In [7], the author formulates a backup deployment model that requires each primary VNF instance to have at least one backup VNF instance, which can improve the availability of functions in the event of a failure. And the author considers the sharing of resources. When network functions fail, resources can be shared and scheduled, which makes SFC more reliable. In [8], authors solve the scheduling problem of SFC by resource constraints and minimizing the backup cost of scheduling. The work in [9] describes the joint problem of VNF scheduling and path selection. The authors consider link constraints and server constraints as guidelines for deploying SFC. But they do not consider end-to-end latency and service reliability. In [10], Bari et al. take the lead in proposing the VNF orchestration problem, which includes VNF deployment and routing problems. The author assumes that all nodes in the network can deploy VNF instances, and the author considers some constraint variables to describe the deployment process. In [11], Kang proposed a placement model for backup VNFs. The author used software-defined networking (SDN) to count the availability of primary and backup instances and introduced a genetic algorithm to estimate the availability of instances and the number of physical nodes per time slot to increase service reliability. In [12], the function decomposition method was proposed by Long Qu et al., they decompose the VNF and enable multiple nodes. All VNF backups are placed on the link. Although this improves the speed of scheduling VNFs, it not only increases the bandwidth resources of virtual links but also greatly increases network energy consumption. In [13], the authors consider reliability-aware SFC placement with the goal of minimizing total resource usage costs, including physical server and physical link costs. However, the author did not consider the problem of virtual resources and lacked consideration of parameters such as delay. In [14], authors improve the reliability and security of SFC by adding processes and signatures that enforce ordering. But in fact, in order to improve the reliability of SFC, it is often necessary to increase the number of infrastructures to increase the cost. In the above work, a large number of backup VNFs will not only occupy additional computing resources of nodes but also frequently enable or disable backup VNFs will also generate higher energy consumption. These methods do not consider the network status of the backup node and the specific backup selection scheme. Therefore, a new SFC orchestration architecture is urgently needed to solve the above-mentioned problems of the VNF backup architecture.

B. INDICATORS OF SFC

Researchers often use indicators such as latency and bandwidth consumption as target parameters for VNF placement and SFC orchestration problems. In [15], some target parameters to be considered when placing ordered VNFs are proposed. For example: minimize the number of active nodes, minimize end-to-end latency, and minimize energy consumption or bandwidth. In [10], the author considers deployment cost and traffic forwarding cost and considers VNF deployment as a problem of minimizing operational cost. But the author did not consider service reliability and forwarding delay. In [16], the authors set the goal of the SFC deployment problem to minimize server and link costs. The authors consider minimizing resource consumption to minimize operating costs and dynamically provision new services to maximize revenue for network operators. Yu [17] proposes an optimal placement method for the problem of resource allocation in SFC, which takes the throughput of network functions as an indicator and establishes a predictive model of the network. This method prioritizes VNF instances with better performance. Some researchers focus on considering energy consumption and delay as performance indicators for evaluating SFC. In [18], the authors consider the perception of energy consumption and the perception of latency. This solution can effectively reduce the energy consumption of servers and switches during VNF deployment. Although the authors considered energy consumption and latency, reliability was not considered. In [19], SFC placement is considered to minimize latency. But this work does not consider the reliability of SFC and the energy consumption during deployment. None of the above papers consider the dependencies between VNFs. In fact, the deployment problem of SFC is different from the traditional VNF deployment problem. Since traffic needs to pass through SFC, it must pass through multiple VNF instances, and there are dependencies between these VNF instances. So we have to place VNF instances in a specific order. Therefore, the dependencies between VNFs are considered in our model. In the process of designing SFC, how to map VNF nodes and links while considering the dependencies between VNFs is an important issue [20]. A dependency is when nodes need to have a prescribed order so that traffic can be routed correctly. Most of the above work only focuses on VNF deployment and backups each VNF in the SFC, which sometimes wastes too many backup instance resources. They do not consider routing paths. In this article, we selectively deploy primary and backup instances to save server capacity and find routing paths for flows.

C. MODEL OF SFC

In the above work, the author first builds a mathematical model, and then formulates an orchestration scheme for a single parameter, or selects some specific parameters for different types of streams and ignores others, which is incomplete. Furthermore, the authors only consider the cost model when considering the architecture, and lack research on the SFC flow request model. Some people consider using the characteristics of SDN to improve the SFC orchestration architecture to improve the deployment efficiency of VNF. In [21], SHARMA proposes a network model with optimization constraints. The model analyzes the availability of over-the-top (OTT) services and improves the operator's profit while meeting the availability. However, this model lacks the consideration of backup and delay. In [22], the authors propose a distributed architecture that can improve the robustness of SFC by screening each VNF and physical nodes through a fully distributed asynchronous consensus mechanism and priority strategy. In [23], the authors propose a prioritized SFC deployment problem and propose a corresponding architecture. Through advanced architecture and algorithms, this method can better meet the QoS requirements of services. Eramo proposes an asymmetric LSTM traffic prediction procedure. This scheme reduces the cost by 40% compared to the classical LSTM prediction method [24]. In [25], authors propose an innovative resource allocation framework for virtualized network environment based on artificial intelligence technology application. Experimental results show that this scheme significantly outperforms non-integrated classical solutions that perform capacity prediction and allocation processes separately. To combine SDN with SFC, Dwaraki et al. proposed an adaptive service-chain routing algorithm to solve this problem [26]. In this algorithm, the authors convert the network graph into a hierarchical graph. The flow with SFC requests is then routed by executing the traditional SP algorithm on the hierarchical graph, obtaining the path with the smallest end-to-end delay. Moens and De Turck first divided physical nodes into computing nodes and dedicated hardware devices in [27]. They treat VNF instances of the same type as a whole, and then they share VNF instances between services to improve VNF utilization. However, the author did not formulate a complete node selection scheme, and also lacked a selection method for multiple VNF instances within a node. Furthermore, this method lacks modeling between VNF instances. In our study, we built a hierarchical network model to construct SFC. We also treat VNFs of the same type individually as a whole, but we treat them as a VNF layer, and we select VNF instances by availability metrics and assign them to physical servers that meet constraints. The closest thing to our study is [28]. The author considers the model problem of the construction of SFC. The author constructs SFC as an aggregated shared functional graph, and then perform reliability screening on VNFs in the atlas. In the routing problem, the author divides the algorithm into two stages to route the SFC flow. First, he selects the appropriate VNF instance, and then connects the instances in the second stage. When routing traffic, both VNF deployment and path selection can affect overall network performance.

TABLE 1. Literature research comparison.

Literature	Reliability (Y/N)	Delay (Y/N)	IT resources (Y/N)	Architecture (Y/N)
[6]	Y	Y	Y	N
[7]	N	Y	Y	Y
[8]	N	Y	Y	N
[10]	Y	N	Y	N
[11]	Y	Y	Y	N
[12]	Y	Y	N	N
[13]	Y	N	Y	Y
[14]	N	Y	Y	N
[16]	Y	Y	Y	Y
[20]	Y	N	Y	Y
[21]	N	N	Y	Y
[22]	N	Y	Y	N
[25]	N	Y	Y	Y
[26]	Y	Y	Y	Y
[27]	N	Y	Y	N
[29]	N	Y	Y	N
[30]	N	Y	Y	Y
[31]	N	Y	Y	N

D. ALGORITHM OF SFC

For the SFC arrangement problem, many researchers have proposed solving algorithms. In [29], credibility-based deployment strategy (CBDS) builds a hierarchical credibility model, using the analytic hierarchy process to select nodes with higher reliability and availability in the hierarchical network. Regarding the problem of how and where to place VNFs and deploy SFCs in an orderly manner, [30] divides them into two categories, which are based on ILP and based on greedy heuristics. In [31], the authors propose an RA-RA routing algorithm. The algorithm first establishes a mathematical model of the BIP, considering the CPU utilization, delay, and bandwidth of the node. Then use the K shortest path algorithm to select VNF nodes. After selecting the VNF node, iterate through K paths in turn to calculate the routing path that meets the requirements. In [32], Nguyen proposed an algorithm to predict actual demand using uncertain demand knowledge. By dividing the delay into multiple steps to calculate and predict, the SFC deployment scheme that meets the requirements is finally obtained. Chen proposed the single service chain solution with heterogeneous vNFs (SEV) algorithm in [33], which considers the resource capacity and traffic ratio to optimize the deployment of VNFs. Gao et al. [34] propose a simple and efficient online algorithm which retains routing decisions when no flow requests come in. Farkiani used the Benders decomposition algorithm in [4] to simplify the SFC deployment problem and proposed a fast and scalable polynomial time algorithm called PDDB to find the energy optimal solution. Most of the above articles use some iterative or heuristic algorithms, which make the solution easy to be limited to the local optimal solution or increase the complexity of the solution.

III. NETWORK MODEL AND PROBLEM DEFINITION

According to the survey of related work, the hierarchical approach can schedule network resources more efficiently and construct SFC. When there are too many virtual resources and physical resources in the network and the topology is complex, we can divide the resources that meet the conditions through layers. In addition, a regular network structure is more suitable for routing the traffic requested by the service. Traditional network architecture citations lack the sequential deployment of services when there are many service requests in the network. The layering is because virtual resources and physical resources are layered, which reduces the coupling degree between network resources. In addition, a layered network also makes it easier to find the underlying host for multiple VNF instances in the upper layer. In this section, we propose a network model consisting of a physical network, virtual network, SFC and hierarchical graph. We divide physical and virtual networks into hierarchical topologies. Meanwhile, we model SFC requests as a hierarchical directed graph. We build a suitable layered network topology that combines the architectures of the virtual layer and the physical layer, we select VNFs in the virtual layer, and the physical layer is responsible for the placement of the VNFs.

A. SYSTEM MODEL

1) PHYSICAL NETWORK LAYERING SCHEME

The physical network consists of underlying physical nodes, and there is a physical connection between every two adjacent physical nodes. The physical network is often used to deploy VNF instances. The physical network includes servers, switches, and routers [35]. Layered diagrams [36] are tools for modeling SFC. Since the nodes in the SFC have a certain order, there is a more definite order relationship between the corresponding physical network nodes. Therefore, in order to correspond to the sequential characteristics of SFC, we layer the physical network. We represent the underlying physical network as a hierarchical undirected graph $G_p^L = (V_p^L, E_p^L)$. Among them, L is the set of various layers of the underlying network, l_n represents the layer n in the hierarchical graph, $L = \{l_1, l_2, \cdots l_\xi\}$. We stipulate that VNFs of different types of functions must be mapped to servers of different physical layers. V_p^L is the set of underlying physical nodes, $n_p^{l_n,i}$ represents the node i of layer n, $n_p^{l_n,i} \in V_p^L(1 \leq i \leq k)$. E_p^L represents the set of underlying physical links, $l_p^{n,(i,j)}$ represents the physical connection between node i and node j, $l_p^{n,(i,j)} = (n_p^{l_n,i}, n_p^{l_n,j}) \in E_p^L(1 \leq i \leq j \leq k)$. $|V_p^L|$ is the number of underlying nodes (the number of vertices), and $|E_p^L|$ is the number of connected edges (the number of links). Each physical node has certain physical computing resources and a certain capacity. We use C_i to denote the capacity of node $i \in V_p^L$. Each link has a certain bandwidth resource, and $C_{i,j}$ represents the bandwidth resource capacity of the edge $(i,j) \in E_p^L$. In order to simplify the model, we assume that the node capacity of the source node and the destination node is not considered when considering the edge capacity, we let $e \in E_p^L$ to represent the original edge, and the capacity of the edge is $C(e)$.

2) VIRTUAL NETWORK LAYERING SCHEME

The virtual network consists of virtual nodes, and virtual nodes are composed of virtual machines that host VNFs. We assume that each virtual machine can only host one type of VNF. Similar to physical networks, we represent virtual networks as undirected graphs $G_v^L = (V_v^L, E_v^L)$. Among them, V_v is the virtual node-set, n_v^i represents the virtual network node i, $n_v^i \in V_v(1 \leq i \leq k)$. E_v is the virtual link set, $l_v^{i,j}$ represents the virtual link between node i and node j, $l_v^{i,j} = (n_v^i, n_v^j) \in E_v(1 \leq i \leq j \leq k)$. Our planned virtual network supports the layered model of SFC. In our hierarchical model, VNFs of the same functional type are in the same layer, and virtual nodes are connected across multiple virtual layers, so our virtual links also logically connect multiple VNFs in abstraction layers. Furthermore, we take advantage of the inherent sharing properties of virtualized resources to improve the efficiency of resource backup.

3) SERVICE FUNCTION CHAINING MODEL

We model a single SFC request as a hierarchical directed graph $G_{SFC} = (src, dst, F, D_v^n, D_v^l)$, where src and dst are the source and destination nodes of the SFC, $src, dst \in V_v$. F represents the ordering of multiple VNFs requested by an SFC, each VNF representing a specific network function, such as firewall, deep packet inspection. There is a certain order of connection between these VNFs, $F = \{F_1^{k_1}, F_2^{k_2}, \cdots, F_n^{k_n}\}, n \in \mathbb{N} > 0$; Num_G indicates the number

TABLE 2. List of parameters with their definitions.

Parameters	Description
L	Set of network layers
p	Physical network
v	Virtual network
G_p^L	Hierarchical graph of a physical network
V_p^L	Collection of physical nodes
E_p^L	Set of physical links
G_v^L	Layered graph of a virtual network
$n_p^{l_n,i}$	The ith physical node of n layer
$l_p^{n,(i,j)}$	Link between nodes i, j
$C_{i,j}$	Bandwidth resource capacity between nodes
C_e	Bandwidth resource capacity
n_v^i	Virtual node
$l_v^{i,j}$	Virtual link
G_{SFC}	SFC request model
F	The ordered set of all VNFs at n layers
Num_G	Number of SFCs
D_p^n,D_p^l	Resource requirements for physical nodes and links
A_v	Availability
F_i^p	Active node i at layer p
R_p^u	State of the physical node u
$\Pr_u^L$	Priority index of node u in the layer L
Re	Product of VNFs availability

TABLE 3. List of decision variables with their definitions.

Variables	Description
$x_{jj'}^i$	Dependency variable
$a_{jj'}^i$	Dependency constraint Variables
$x_{n_p^{l_c,g}}^{n_v^i}$	Cth layer physical node g hosts the virtual node i
$y_{l_p^{m,n}}^{l_v^{i,j}}$	Physical link (m,n) carries the virtual Link (i,j)
$l_{F_i^p,F_j^q,v,w}$	Mapping direction variable of (v,w)
$q_{F_i^p,v}$	Mapping relationship between active node i and v

of SFCs, $F_i = \{F_i^1, F_i^2, \cdots, F_i^{k_i}\}, i \in \mathbb{N} > 0$, which $F_i^k \in F_i$ indicates that F_i^k is the kth VNF instance selected from the ith layer VNF, F_i represents the layer ith VNF set, and the function types of the VNFs of this layer are the same. D_v^n and D_v^l respectively Resource requirements for VNF nodes and virtual links.

B. PROBLEM MODEL

As shown in Figure 1, each flow has a specified entry node *Src*, an exit node *Dst* and a SFC. SFC contains several VNFs that satisfy dependencies. In the example shown in Figure 1, the order of SFC assignment is $VNF_1 \rightarrow VNF_2 \rightarrow VNF_3 \rightarrow VNF_4$. Traffic must enter the network through the ingress node, pass through firewalls, deep packet inspection, encryption and decryption, and finally exit the egress node. If the order is not followed, the traffic demand cannot be met. There are multiple physical servers at the physical layer, and these servers can host virtual machines. We assume that a physical machine only hosts one VNF. We refer to virtual machines as virtual nodes and physical machines as physical nodes, so the order of physical nodes is $Server_1 \rightarrow Server_2 \rightarrow Server_3 \rightarrow Server_4$. Due to the large number of

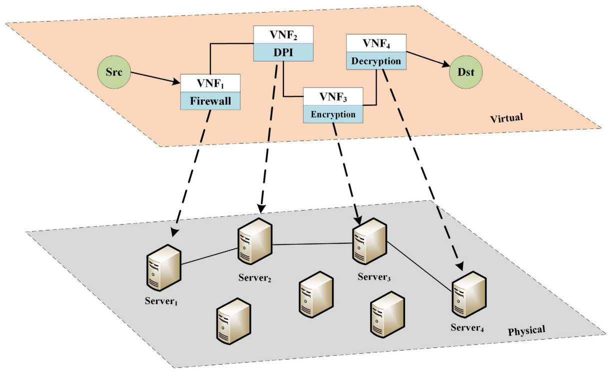


FIGURE 1. SFC deployment process.

nodes and the complex network environment, each virtual node may host multiple VNFs, and each physical node may host multiple virtual nodes. This makes some nodes have higher load rates. If the selected node fails, the path will also fail, which will cause traffic to be routed correctly, which greatly wastes network resources. Therefore, we need efficient network models and algorithms to find suitable physical nodes and paths. In SDN, the controller gets a global view of the network. Therefore, we utilize the SDN controller to collect network status and manage the network. When a service request comes, we need to select the placement path of the VNF according to the current network state. Furthermore, the order of VNFs in the SFC must be maintained.

C. MATHEMATICAL MODEL

1) PHYSICAL NETWORK LAYERING SCHEME

Each physical server and its neighbors are grouped together in a physical network. Since SFCs have a specific order, each server or node group also has a certain mapping order. Because of this order constraint, we layer the physical network as shown in Figure 2, where we use a simplified model, replacing servers with physical nodes. Physical nodes are divided into master nodes and backup nodes according to the mapping requirements, and the backup node is the adjacent node of the master node, and the backup node may also have its own backup nodes. Besides, the physical layer is also divided into an initial layer and a backup layer. If all master nodes can meet the mapping requirements during the SFC deployment process, there is no need to search for backup nodes in the backup layer. However, the network environment is often complex, so most master nodes face high load problems, so the use of backup nodes is unavoidable. We use the SDN controller to manage the underlying physical network to achieve hierarchical division of the network and forwarding of instructions.

2) VIRTUAL NETWORK LAYERING SCHEME

In our SFC-HO, VNFs of the same functional type occupy a layer in the virtual network, and this layer is identified according to the functional type of the VNF. Since there

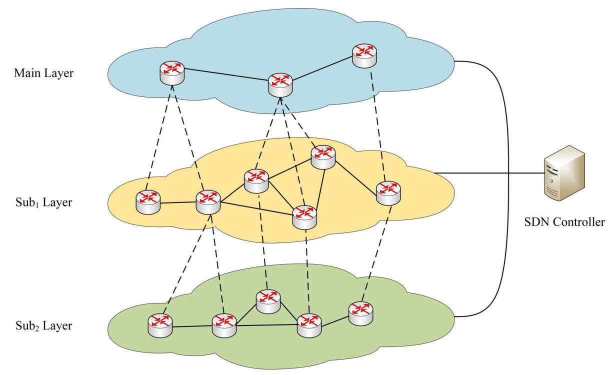


FIGURE 2. **Physical network hierarchical model.**

are multiple VNF instances with the same function on a computing node, we divide the nodes according to the same type of VNF instances. For example, a firewall layer, a deep packet inspection layer, etc. may exist in the virtual network. Although the functions of VNFs in a single layer are the same, the status of each VNF in each layer is different. We assume that a virtual node only hosts one VNF. We introduce the VNF availability indicator A_v here. Since we assume that a VNF corresponds to a virtual node, the VNF availability can also be expressed as the availability of virtual nodes. That is, we describe the VNF availability by the normal power-on state indicator of the virtual machine.

$$A_v = \frac{MTBF}{MTBF + MTTR} \tag{1}$$

In (1), *MTBF* is the mean time between failures of the virtual machine, and *MTTR* is the mean time to repair of the virtual machine. We use the relationship between them here to represent availability.

An SFC will contain multiple VNFs. When an SFC request arrives, the network operator determines which functional types of VNFs are required according to the request. Due to the layered mechanism of the SFC we built, our selection of VNFs will be divided into two steps:

- Filter VNFs of the same function type and assign them to different virtual layers according to the function type. Each layer can only have VNFs of one function type.
- At each layer, virtual nodes and the VNFs they host are selected based on availability. The higher the availability, the higher the priority. The higher availability VNF is chosen to improve SFC reliability.

3) PHYSICAL NETWORK AND VIRTUAL NETWORK

To describe the relationship between nodes and links in physical and virtual networks, we introduce two variables $x^{n_v^i}_{n_p^{l_c,g}}$, $y^{l_v^{i,j}}_{l_p^{m,n}}$. And $n_p^{l_c,g}$ denotes the node g at layer c in the physical network, where $c \leq \xi$, $g \leq k$. When this physical node g hosts the virtual node i, $x^{n_v^i}_{n_p^{l_c,g}} = 1$, otherwise $x^{n_v^i}_{n_p^{l_c,g}} = 0$. $l_p^{m,n}$ represents a physical link between physical nodes m and n, $m, n \leq k$. When the link $l_p^{m,n}$ carries a virtual link $l_v^{i,j}$, $y^{l_v^{i,j}}_{l_p^{m,n}} = 1$, otherwise $y^{l_v^{i,j}}_{l_p^{m,n}} = 0$.

In our hierarchical model of SFC, when a service request arrives, the SDN controller filters out the required VNF types according to the service request, and then selects VNFs based on availability metrics. Since each type of VNF is associated with the corresponding layer, we only need to select a VNF from the VNF layer that satisfies the constraints, and the VNFs in this layer are in a backup relationship with each other. After screening the VNFs of each layer, each selected VNF is connected to form an SFC.

In layer i, we denote the availability of layer i VNFs with A_v^i, and we choose VNFs with the highest availability:

$$A_v^i = Max\{A_i^1, A_i^2, \cdots, A_i^k\} \tag{2}$$

When all VNFs are selected, the ordered set of VNFs can be expressed as:

$$VNF_{set} = \{VNF_1, VNF_2, \cdots, VNF_j\} \tag{3}$$

We connect the sequential set of VNFs with source and destination nodes to form SFC to improve the reliability of SFC. We aim to provide users with more reliable and diversified services.

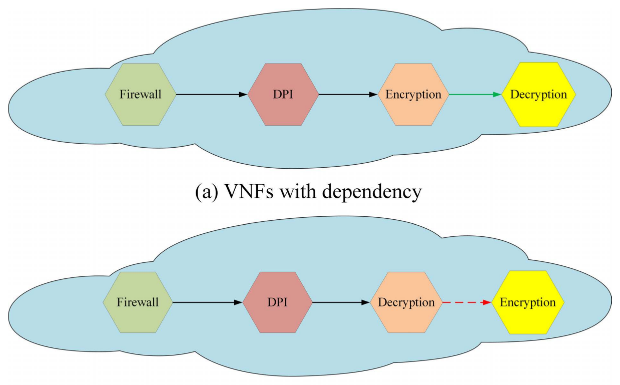
IV. SFC-HO

In this paper, we propose a hierarchical orchestration method for SFC. This method first divides the physical network into layers according to the previous method, and then this method divides VNFs according to function types. We route traffic using designated service function paths, which are necessary to provide services through SFCs. We then demonstrate the strengths of our orchestration approach through the effect of delivering services. This traffic has passed the specified network function, and we evaluate our solution by metrics such as request acceptance rate and latency.

When an SFC request comes, we model the SFC as a hierarchical graph, and we use the global capabilities of SDN to obtain the state information of each node in the network. We deploy our VNF instances hierarchically based on the state of the nodes. Finally, we consider the routing problem and solve the complete SFC routing process. SFC should consist of VNFs that satisfy dependencies. To describe possible conflicting relationships, We use the matrix $A^i = [a^i_{jj'}] \in \{0, 1\}^{L_i \times L_i}$ to represent the dependencies between VNFs in S_i^*. S_i^* is the collection of all VNFs. A^i can be derived from the matrix A according to the VNF type. We further use the matrix $X^i = \left[x^i_{jj'}\right] \in \{0, 1\}^{L_i \times L_i}$ to represent the composition result. $x^i_{jj'} = 1$ means that the dependencies between VNFs are satisfied. Finally, (4) is derived to ensure that the SFC combination result satisfies the dependencies between VNFs.

$$x^i_{jj'} - a^i_{jj'} \geq 0, \forall i, j, j'. \tag{4}$$

As shown in Figure 3, there are two SFCs connecting VNFs of the same type. We assume that they provide a service with



(a) VNFs with dependency

(b) VNFs with no dependency

FIGURE 3. The dependency of VNFs.

a security function. In Figure 3(a), the traffic passes through the four functions of firewall, packet inspection, decryption and encryption in sequence, forming an SFC with the correct sequence of functions. However, in Figure 3(b), the SFC first decrypts and then encrypts in the orchestration process, which not only leads to logical errors but also makes the traffic unable to be routed correctly and cannot provide confidential services. Therefore, the dependencies between VNFs must be considered in the SFC orchestration process.

Taking Figure 4 as an example, an SFC contains four types of VNFs, namely firewall, deep packet inspection, encryption and decryption. First, the firewall needs to be mapped to a physical node or host at the first layer. However, the reliability of the corresponding host does not meet the deployment requirements, so we select the second-layer node corresponding to this node to continue deploying the VNF instance. After successful deployment, it is reconnected to the first layer according to the topology relationship between nodes, and then the second VNF instance is mapped. Subsequent operations are the same. But it is worth noting that if all the nodes in the second layer do not meet the reliability requirements, continue to the next layer until a physical host that meets the deployment requirements is found. In the process of deployment, it must be considered that the traffic can pass through the ingress and egress nodes. In addition, traditional network models often difficult to obtain a hierarchical network graph view, and it is difficult to manage multi-layered networks. SDN not only has the ability to obtain a complete network view but also can centrally manage and control nodes in the network. We use the SDN controller to issue instructions, which enables more flexible connections between nodes at each layer. We cleverly use the centralized management capabilities of SDN to improve the layered network topology and improve the intelligence of our solutions. In our model, low-reliability servers are not used to deploy VNFs, but they serve as connection points between nodes. When the SDN controller issues a list command, these nodes only forward according to the list, and do not need to bear the VNF separately. That is, if the primary node of the first tier is unreliable, it acts as a bridge between the previous server hosting the VNF instance and its own backup node that satisfies the VNF instance. The paths between the hosts are solved by our proposed RAHR algorithm. In this process, the routing algorithm runs in the router of the physical layer, and the specific path is determined by the routing table and the forwarding table. The SDN controller only handles the placement of VNFs in this process. We inevitably have to consider the problem that the controller may be faulty. When one controller fails, the controller sends commands to the backup controller, which then takes over. These controllers may be standby units on standby, or they may be the controllers that manage the neighborhood.

Delay is an important indicator to be considered in the SFC orchestration process. When calculating network delay, there are generally four kinds of delays, namely node processing delay, queuing delay, sending delay and propagation delay. For the convenience of processing, here we consider the processing delay of the node and the transmission delay of the link. We express the delay as the processing delay of the physical node and the transmission delay of the link. Here we do not consider the source and destination nodes of the SFC, so we can simplify the link between m and n to e. The end-to-end delay of the entire chain is equal to the sum of the processing delay of all physical nodes plus the transmission delay of each link.

$$\begin{aligned} DELAY = &\sum_{n_v^i \in V_v} \sum_{n_p^{l_c,g} \in V_p^L} x_{n_p^{l_c,g}}^{n_v^i} D(g) \\ &+ \sum_{l_v^{i,j} \in E_v} \sum_{l_p^{m,n} \in E_p^L} y_{l_p^{m,n}}^{l_v^{i,j}} D(e) \end{aligned} \tag{5}$$

$D(g)$ and $D(e)$ are the processing delay of the physical node and the transmission delay of the link, respectively. The delay in (5) does not consider the queuing delay and link failure of the link. Our research focuses on improving reliability while keeping latency as low as possible.

In [9], the authors have given sufficient proof that the deployment of SFC is an NP-hard problem. In addition to this, we consider the link selection problem, so our SFC deployment problem is also NP-hard. D_p^n and D_p^l are the bandwidth resource requirements of physical nodes and physical links, respectively, where $l = (i, j)$. The bandwidth consumption of the entire link is equal to the sum of the bandwidth consumption of all deployed paths.

$$\sum_{(i,j) \in V_p^L} D_p^{(i,j)} \leq \sum_{(i,j) \in V_p^L} C(i,j), F_{i,j} \in F \tag{6}$$

Constraint (6) represents the link bandwidth constraint, and the constraint is used to ensure that the bandwidth of the mapped link does not exceed the bandwidth capacity of the link. If the bandwidth required for mapping is higher than the bandwidth capacity, the SFC cannot be mapped. In addition to the constraints of the bandwidth of the link, the node also has a certain resource capacity.

We use C_u to represent the capacity of the physical node u and D_p^u to represent the maximum resource demand of the physical node u.

$$D_p^u \leq C_u, u \in V_p^L \tag{7}$$

The node capacity constraint in (7) is used to ensure that the computing resource requirement of the mapping node does not exceed the maximum resource requirement of the node. When the node capacity constraints are not sufficient to support the deployment, other nodes will be chosen to map the VNF.

$$x_{n_p^{l_c,g}}^{n_v^i} \in \{0,1\}, \forall n_v^i \in V_v, n_p^{l_c,g} \in V_p^L \tag{8}$$

$$y_{l_p^{m,n}}^{l_v^{i,j}} \in \{0,1\}, \forall l_v^{i,j} \in E_v, l_p^{m,n} \in E_p^L \tag{9}$$

Constraint (8) and constraint (9) are the value ranges of the mapping variables, respectively. $x_{n_p^{l_c,g}}^{n_v^i} = 1$ indicates that the physical node g hosts the virtual node i. $y_{l_p^{m,n}}^{l_v^{i,j}} = 1$ indicates that the link $l_p^{m,n}$ carries the virtual link $l_v^{i,j}$.

$$x_{n_p^{l_c,g}}^{n_v^k} + x_{n_p^{l_c,g}}^{n_v^m} \leq 1, \forall k \in V_v, \forall m \in V_v, \forall g \in V_p \tag{10}$$

Constraint (10) means that each virtual node can only be mapped to one underlying physical node, therefore, each VNF can only be mapped to one underlying physical server. Due to the existence of the network layering model, although the nodes in a network layer may interfere with each other, the backup node of a node may be another node in the same layer. In order to simplify the model of the problem, we assume that each backup node does not interfere with each others, and there are at least two intermediate nodes between each master node.

$$\prod_{i,g \in k, c \in \xi} x_{n_p^{l_c,g}}^{n_v^i} = 1, \forall F_{i,g} \in F \tag{11}$$

Constraint (11) means that all virtual nodes are mapped to physical nodes, that is to say, VNF instances have corresponding computing resources of physical machines and CPUs.

$$\sum_{n_v^i \in V_v} \sum_{n_p^{l_c,g} \in V_p^L} x_{n_p^{l_c,g}}^{n_v^i} D(g) + \sum_{l_v^{i,j} \in E_v} \sum_{l_p^{m,n} \in E_p^L} y_{l_p^{m,n}}^{l_v^{i,j}} D(e) \leq D_{\max} \tag{12}$$

Constraint (12) is the delay constraint of the service path, and D_max is the maximum tolerated delay of the service path.

We consider link mapping variables to ensure that paths between adjacent functions are mapped to adjacent physical paths. Besides, due to the complexity of the network model, traffic reversal may occur. Therefore, we need to consider not only the mapping direction of the VNF but also the direction of the traffic. We let $l_{F_i^p, F_j^q, v, w}$ be the link map variable. If the routing path between two adjacent functions traverses part of the physical network link $l_{v,w}$, and the traffic traverses endpoints v and w, $l_{F_i^p, F_j^q, v, w} = 1$, $l_{F_i^p, F_j^q, v, w} = 0$.

$$\sum_{w \in V_p^L} l_{F_i^p, F_j^q, v, w} - \sum_{w \in V_p^L} l_{F_i^p, F_j^q, w, v} = \begin{cases} 1, q_{F_i^p, v} = \wedge q_{F_j^q, v} \neq 1 \\ -1, q_{F_j^q, v} = \wedge q_{F_i^p, v} \neq 1 \\ 0, \quad else \end{cases}$$
$$\forall v \in V_p; \forall F_i^p, F_j^q \in F \tag{13}$$

Equation (13) is a routing constraint to ensure the connectivity between the mapped physical servers. $q_{F_i^p, v}$ represents the mapping relationship F_i^p with physical network nodes v.

We consider the computing resources of the CPU. $s(u)$ represents the CPU computing resources of the node, and $bw(i,j)$ is the link bandwidth between nodes i and j. Therefore, we use (14)(15) to denote the computational resource cost of physical nodes and the bandwidth resource cost of physical links.

$$c_u^{cpu} = \frac{\max_{u \in V_p^L} s(u)}{C_u - D_p^u} \tag{14}$$

$$c_{(i,j)}^{bw} = \frac{\max_{(i,j) \in E_p^L} bw(i,j)}{C(i,j) - D_p^{(i,j)}} \tag{15}$$

Our cost function is set as the sum of the computational resource cost of all physical nodes mapped by the SFC plus the sum of the bandwidth resource cost of all links.

$$COST = \sum_{u \in V_p^L} c_u^{cpu} + \sum_{(i,j) \in E_p^L} c_{(i,j)}^{bw} \tag{16}$$

In our scheme, we strive for the highest SFC reliability, which we define Re as the product of the availability of all VNFs. The goal is to minimize the quotient of cost and reliability.

$$\mathrm{Re} = \prod_{i=1}^{j} A_v^i \tag{17}$$

In our scheme, the product of the availability of each node is the SFC reliability, and (17) is our formula. To strike a balance between cost and reliability, we formulate a reliability constraint:

$$\mathrm{Re} \geq 0.3^j \tag{18}$$

Expression (19) is our objective function, and we are committed to improving reliability and reducing costs under the premise of satisfying the previous constraints.

$$\min \frac{COST}{\prod_{i=1}^{j} A_v^i} \tag{19}$$

Figure 4 shows our proposed SFC layered orchestration architecture. Taking the SFC orchestration process in the figure as an example, we represent the hexagon as VNF, and the functions of these virtual layers will be mapped to the underlying

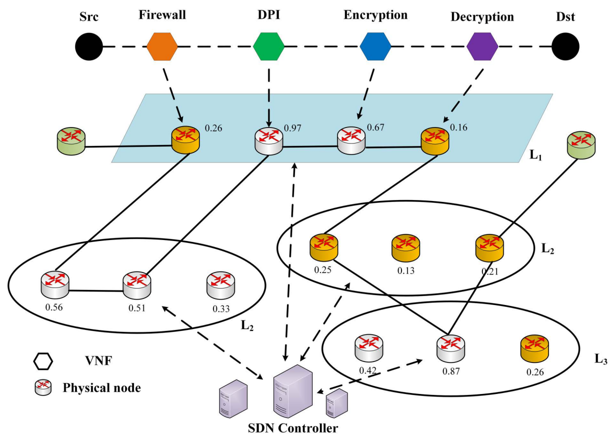


FIGURE 4. SFC hierarchical orchestration.

physical network. The underlying physical network is divided into several layers as shown in the figure. Each layer contains several nodes. The physical nodes on the same layer are in a backup relationship, and the nodes are mutually backup, and each node has a different physical state. We consider the node reliability to represent the state of the physical node u, and the reliability is represented by R_p^u.

In order to more vividly represent the reliability of different nodes, the reliability in the figure is represented by numbers. We consider the computational cost and load of nodes as evaluation metrics for reliability, and the reliability is represented by (20). We put forward the idea of priority. For example, the reliability of 0.8 1 has the highest priority, and the specific priority is classified in (21). $\Pr_u^L$ represents the priority index of node u in the layer L. During the orchestration process of SFC, we give priority to nodes with high priority. *middle* indicates moderate reliability, and if there is no high-reliability node, a medium-priority node is used to map the VNF. *low* means low reliability, and this type of node is only considered when the rest of the nodes are unavailable. Finally, the reliability of the node may be 0 due to server failure or power failure. When all physical nodes at a certain layer are unreliable, we abandon the VNF mapping, and the SDN controller will notify the network operator of the service response failure. In Figure 4, VNFs such as firewall and deep packet inspection are mapped to L_1 to L_4 layers respectively. After the mapping is completed, they are connected to the source node and the destination node to complete the SFC mapping process.

$$R_p^u = \frac{D_p^u}{C_u \ln c_u^{cpu}} \tag{20}$$

$$\Pr_u^L = \begin{cases} high & 0.8 \le R_p^u \le 1 \\ middle & 0.3 \le R_p^u < 0.8 \\ low & 0 < R_p^u < 0.3 \\ 0 & R_p^u = 0 \end{cases} \tag{21}$$

Through the above methods, we first innovatively establish a perfect physical network layer model and virtual network layer model. Second, we classify and filter VNFs based on the functional type and utilize the criteria of availability. After that, we consider metrics such as latency and cost that are common in physical networks. Besides, we innovatively consider the reliability of SFC and combine it with the total cost. We also consider routing issues and dependencies between VNFs that are often overlooked in many related studies. We reduce total cost while maximizing reliability as much as possible. Our research is ultimately focused on delivering highly reliable SFCs while meeting energy consumption requirements.

V. SOLUTION METHODOLOGY

A. RAHR

In addition to deploying VNFs to physical nodes, the orchestration process of a complete SFC also needs to select paths between servers to route traffic. The RAHR algorithm is based on a layered network model, and we consider the reliability of the SFC as the main indicator for traffic routing. We consider the problem of calculating paths between physical servers after VNFs are deployed to physical servers in sequence. This step-by-step approach can not only improve the deployment efficiency of VNFs, but also reduce route computation time. Therefore, we need a way to decompose our original ILP problem into VNF deployment and link selection problems. We innovatively choose the Benders decomposition method to decompose our problem, which can reduce the complex ILP problem to an approximate solution of the relaxed LP problem to solve the SFC orchestration process.

We use the Benders decomposition method here. Benders generally divide variables into two types. The first is to include variables whose values are set by the first main problem (RMP), which may be an ILP problem. The second includes variables whose values are set by the sub-problems (SP) of the LP problem. The method solves the main problem during iterations and replaces the main problem's values in the subproblems after each iteration. Then solve SP and judge whether the solution is feasible, and the algorithm terminates after reaching the optimal value after many iterations. In traditional SFC orchestration or routing problems, heuristic algorithms and other algorithms often have the problem of easily falling into local optimum, and this method can effectively avoid falling into local optimum. The deployment method of VNF has been described in detail above.

While Benders decomposition allows us to solve two smaller problems rather than one large and complex problem, it may generate infeasible solutions in the initial iteration, so it may take many iterations to find the optimal solution. So while it is more scalable than classic ILP algorithms, it may take longer to solve than them. Furthermore, in this algorithm, given that finding a feasible solution for the ILP problem is NP-hard, the main problem as an ILP problem must be solved in each iteration. We use the Benders decomposition algorithm to decompose the original problem into the main problem master problem and sub-problem subproblem, so that the ILP problem is simplified to the ordinary LP problem.

Original question:

$$\begin{aligned} &Minimize \quad c^T x + f^T y \\ &subject to: \ Ax + By = b \\ &\qquad x \geq 0, y \in Y \subseteq \mathbb{R}^q \end{aligned} \tag{22}$$

(22) is our original SFC-HO problem, where x is the service function path routing problem, and y is the VNF hierarchical mapping problem. Y is the feasible region of y, that is, the deployment scheme that satisfies the deployment constraints of the VNF. A and B are matrices of corresponding dimensions, c, b, and f are vectors of corresponding dimensions. Here, we will treat y as a complex variable. When the value of y is fixed, the original problem becomes a simple LP problem. At this point, we just need to determine the routing path between the two servers. Based on this, we can decompose the problem:

Master Problem:

$$\begin{aligned} &Minimize \quad f^T y + q(y) \\ &subject\ to: y \in Y \end{aligned} \tag{23}$$

Sub Problem:

$$\begin{aligned} &Minimize \qquad c^T x \\ &subject\ to: \quad Ax = b - By \\ &\qquad\qquad x \geq 0 \end{aligned} \tag{24}$$

Expression (23) and expression (24) are the master problem and sub problem after decomposition.

In this paper, we decompose our SFC-HO problem into two parts, first the VNF-hierarchical deployment problem (VNF-HD), and then the routing problem between servers. In the first part, we deploy VNF using our proposed SFC layered orchestration method, fully considering factors such as link bandwidth, node resource capacity, etc., and our goal is to maximize the balance between reliability and minimize cost. In the second sub-problem, we use the enhanced Viterbi algorithm to find the shortest path between servers. We decompose our RAHR algorithm into Algorithm 1 and Algorithm 2 using the decomposition method. Algorithm 1 is a reliability-aware hierarchical deployment (RAHD) algorithm, and Algorithm 2 is an improved Viterbi algorithm.

B. RAHD

We propose the RAHR algorithm to solve the SFC-HO problem. We decompose our SFC-HO problem into a main problem and sub-problems using the Benders decomposition algorithm, the main problem is VNF-HD, and we deploy the VNF using the method described earlier. As mentioned earlier, $q(y)$ is the objective function value of the main problem. Here we assume that the main problem is bounded, because the network topology we choose needs to satisfy constraints, and the model we choose is more flexible. If the main problem is unbounded, it means that there are no virtual nodes or physical nodes that meet the requirements in the network domain, and SFC cannot be orchestrated. As shown in Algorithm 1, we solve the main problem algorithm with the RAHD algorithm. For any given SFC request, we determine the VNF that meets the requirement. After that, we select the appropriate VNF instance in the virtualization layer according to the availability of the VNF. Then, we hierarchically partition the physical network and select effective physical servers to map VNF instances. Finally, we get all the mapping information, and the physical server information has been determined. SDN controller is responsible for processing Algorithm 1, and we will choose the more reliable controller to handle the problem. When one controller fails, the backup controller takes over computing.

$$q(y) = \min \frac{COST}{\prod_{i=1}^{j} A_v^i} \tag{25}$$

In Algorithm 1, we give our designed network model graph, including physical network model and virtual network model. Next, we give the SFC request model and describe the request as a hierarchical directed graph, and we input the corresponding node and link requirements, etc. Finally, we output the SFC deployment variable $x_{n_p^{l_c,g}}^{n_v^i}$, $y_{l_p^{m,n}}^{l_v^{i,j}}$, and service chain Tr. We divide the algorithm into four steps. In the first step, we check the VNFs, determine the dependencies between VNFs according to (4), and discard the VNF types that do not satisfy the relationship. If a given VNF does not satisfy the dependencies, it means that there are fewer types of VNFs provided by the operator, which do not meet the basic traffic forwarding requirements, so we abandon the orchestration of SFC. In the second step, we calculate the reliability of the SFC, we first calculate the availability of the required VNF, and find the VNF instance with the highest availability in the first layer. Then, we traverse multiple VNFs at each layer to find all VNF instances that meet the requirements. During this process, we pick all VNF instances with availability above 0.3 to improve overall reliability. Finally, we examine constraints such as the corresponding bandwidth. In the third step, we model our problem with ILP, and we first add the VNFs that meet the conditions calculated in the second step to the VNF_{set}. Finally, in the fourth step, we deploy the VNF instance. We let the set Tr denote a sequential SFC. Tr contains Src, Dst, and VNF_{set}. Then we calculate the reliability of SFC according to (20). Besides, we again use (4) to determine whether the VNF satisfies the dependencies, and we also use $x_{jj'}^i = 1$ to determine whether the virtual machine is powered on. Next, we add the VNFs that meet the requirements to Tr. Here, we require that the reliability of the SFC must satisfy $\mathrm{Re} \geq 0.3^j$ and $\mathrm{Re} \geq 0.1$, because the lower reliability of a single device will seriously affect the reliability of the overall link. Finally we deploy the SFC according to (12) and (19).

Algorithm 1 RAHD Algorithm

```
Require: $G_p^L, G_v^L, G_{SFC}$
Ensure: $T_r, x_{n_p^{l_c,g}}^{n_v^i}, y_{l_p^{m,n}}^{l_v^{i,j}}$
1:  Set $N_v$, $N_v^w$ to null
2:  for $i = 1$ to $n$ do
3:      Let $A_v^i = Max\{A_i^1, A_i^2, \cdots, A_i^k\}$
4:      if $A_v^i \geq 0.3, i \in [1, k]$ then
5:          Add $n_v^i$ to $N_v^w$
6:      else $i = i + 1$
7:      end if
8:  end for
9:  for all each $n_i$,$n_j \in N_v^w$ do function check
10:     if $x_{jj'}^i - a_{jj'}^i \geq 0, \forall i, j, j'$. then
11:         Add $N_v^w$ to $N_v$
12:     end if
13: end for
14: for $u = 1$ to $k$ do
15:     if $D_p^u \leq C_u$, $u \in [1, j]$ then
16:         Add $n_p^{l_m,u}$ to $N_v$
17:     else $u = u + 1$
18:     end if
19: end for
20: Set $VNF_{set}$ to null
21: for all VNFs, $i, j$ do
22:     Let $\Pr^L = \max \Pr_i^L$, then choose the $VNF_i^L$ which $Pr_i^L = Pr^L$
23:     Add $VNF_i^L$ to $VNF_{set}$
24: end for
25: Let $Tr = \{Src, VNF_{set}, Dst\}$
26: for all VNFs $\in VNF_{set}$ do
27:     compute $R_p^u$ by (20)
28:     if $R_p^u \geq 0.3$ and Re $\geq 0.1$ then
29:         Find the $Tr$ subject to equation (12) and (19), then
30:         Deploy $n_v^i$ to $n_p^{l_m,u}$
31:     end if
32:     Continue to deploy the next $n_v^i$
33: end for
```

C. IMPROVED VITERBI ALGORITHM

In this section, we propose an improved Viterbi routing algorithm. It can resolve the path between each node as determined by the server or host (hereinafter collectively referred to as the physical node).

Algorithm 2 is established based on Algorithm 1. We need Algorithm 1 to deploy the VNF instance and then establish a routing problem based on the physical network topology and traffic requirements. First, we represent the network topology with an adjacency matrix. In addition, the edges in the matrix are represented by service chain paths, and the weights of service chain paths are defined by the priority of (21). Second, we construct a two-dimensional table containing physical nodes and VNF instances. For the deployed VNF instance, we select the nodes between the hosts to form the candidate set of forwarding nodes. Assuming that the SFC flow requires n network functions to be processed sequentially, we form n sets with specific network functions. Furthermore, we discard nodes with reliability lower than 0.1 before applying the Viterbi algorithm. In this way, we can reduce the complexity of the algorithm, the process is shown in Algorithm 2.

Algorithm 2 Improved Viterbi Algorithm

```
Require: $G_p^L, x_{n_p^{l_c,g}}^{n_v^i}, y_{l_p^{m,n}}^{l_v^{i,j}}, len$
Ensure: bestpath
1:  for all $len_k$ in $len$ do
2:      Set $Path$ as null
3:      Find $c_u^{cpu}$, $c_{(i,j)}^{bw}$,$D(g)$ and $D(e)$ for links and nodes
4:      Calculate link and node cost based on equation (14) and (15)
5:      for $j = 1 : k$ do
6:          $n[j+1]$ = the VNF instances according to Algorithm 1
7:      end for
8:      $n[1] = S_i$, $n[k+2] = T_i$
9:      for $j = 2 : k + 1$ do
10:         for $m = 1 : n[j].l_n$ do
11:             if $R_p^j < 0.1$ then
12:                 Remove $n[j][m]$ from $n[j]$
13:             end if
14:         end for
15:         Find $bestpath(j)$ by $Viterbi(j)$
16:     end for
17: end for
```

In Algorithm 2, we perform reliability calculation on the physical nodes mapped by each corresponding VNF instance, then traverse all nodes on the chain. Due to our hierarchical setup, multiple layers need to be traversed as well, which may increase the number of traversed nodes. After traversing a group of nodes, we output the best path *bestpath* (j) between the group of nodes, and finally, form a complete routing path. Experiments show that our algorithm has certain advantages in improving the reliability of the SFC.

D. COMPLEXITY

We use the time complexity of the algorithm to simplify the analysis of the pros and cons of the running time of our algorithm. Our algorithm is divided into two parts, RAHD and the improved viterbi algorithm. RAHD first screens VNFs and adds suitable VNFs to the VNF set. The time complexity of this process is $O(K)$, where K is the number of given VNFs. Then RAHD calculates the dependencies between VNFs, and the time complexity of this step is $O(K(K-1))$. During deployment, the time complexity of RAHD is $O((k-1) \cdot n \cdot (n-1))$. Wherein, k represents the number of VNFs required for the request, that is, the number of times of VNFs that need to be mapped to the physical server. n is the number of states, that is, the number of times the path is calculated between every two nodes.

Since we do not consider the source and destination nodes, the number of mappings can be reduced by one computation. Combining these results, the time complexity is of RAHD is $O\left((k-1)\cdot n\cdot(n-1)\right)$. The time complexity of the Improved Viterbi algorithm is $O\left(n^2\cdot(k-1)\right)$.

VI. SIMULATIONS

We conduct related experiments to evaluate the performance of our algorithm. We first introduce the simulation environment, and then we introduce some important performance parameters. Finally our simulation results. In our simulations, we first compare the performance of SFC-HO with well-known algorithms for solving the ILP problem, i.e., NRCR [28], CBDS [29] and the standard Dijkstra algorithm. Node-ranking algorithm with centrality and reliability(NRCR) focuses on backup nodes and the selection and deployment of backup instances. First, multiple SFCs are aggregated into a service function graph(SFG), and the reliability of the SFG graph is screened. Improve reliability of SFCs that do not meet requirements. The credibility-based deployment strategy(CBDS) builds a hierarchical credibility model, using the analytic hierarchy process to select nodes with higher reliability and availability in the hierarchical network.

A. SIMULATION ENVIRONMENT

To verify the algorithm performance, simulations were performed using MATLAB 2018b software on a desktop computer configured with an Intel Core i7-6700 CPU, 16GB RAM. Referring to the research of [37], the simulation topology adopts the pdh topology on SNDlib, including 32 hosts and 20 switches, and the initial processing capacity of each node is 256units. The experimental parameters are shown in Table 4. In the experiment, each undirected edge of the topology is set as two directed edges, each direction link bandwidth is 700Mbit/s. We set the length of the SFC to be a uniform distribution of 4-6. The resource capacity and resource requirements of nodes obey the uniform distribution of [15, 30] and [5, 10], respectively. The reliability of the controller we selected is [0.9, 0.999]. We compare the processing time of the algorithm, the acceptance rate of service requests, computing resource utilization, bandwidth resource utilization, overall delay and overall cost under different SFC request flows. The infrastructure used in the simulation includes routers, switches, servers, and ingress and egress points. We randomly selected two switches as core switches and matched their ingress and egress nodes. The rest of the switches and routers are laid out in the network according to the structure of the classic USNet topology. For each service, we randomly select ten virtual network nodes that can be used to host VNFs. In the physical network, we select ten more physical nodes, which represent the locations where the VNF is instantiated. When studying our method, we divided 32 hosts into four physical layers according to the combination of 5, 12, 8, and 7. We divide them into one main layer and three sub-layers. For other research methods, we distribute network nodes according to USNet.

TABLE 4. Simulation parameters.

Parameters	Value
Bandwidth	$700Mbps$
Node resource capacity c_i^{cpu}	Uniform [15-30]
Node resource demand D_i	Uniform [5-10]
Chain length	Uniform [4-6]
Number of switches	20(USNET)
Number of servers	32(USNET)
Controller reliability	[0.9,0.999]

We use randomly generated service requests. For each service request G, we randomly choose the number of requests k_G, $1 \leq k_G \leq Num_G$. The connections of these VNFs need to satisfy dependencies, and the traffic demand between each function follows a uniform distribution [1, 5]. For service requests, we generate each VNF sequence based on the samples of [38]. Its number is randomly distributed in [38] according to a Poisson distribution, with an average arrival rate of 1 per 20-time units. The average duration of each request of 500-time units, following a negative exponential distribution.

B. SIMULATION RESULTS AND ANALYSIS

Simulation Results and Analysis We compare our algorithm with three known algorithms, namely NRCR, CDBS, and Dijkstra. Given 5-100 SFC flow requests, we conduct experiments on algorithm running time, service request acceptance rate, CPU resource utilization, bandwidth resource utilization, total delay, computing resource cost, bandwidth resource cost, and total cost. We set the iteration number distribution for NRCR algorithm and CDBS to 200 and 20, and the iteration number to 50 for RAHR. We consider the performance of our algorithm with traffic rates of 20% and 60%, respectively, so that we can get a clearer picture of the advantages of our method.

The results of the algorithm running time are shown in Figure 5 and Figure 6. The icon indicates the processing time to find the best solution. CBDS is very similar to our method in that it first builds a hierarchical model. But as the number of requests increases, its processing time decreases relatively. Because CBDS uses a simpler path calculation method for the physical network, this results in the shortest total time. When service requests increase, CBDS will converge quickly. NRCR is divided into four parts. The second part uses the full permutation to calculate a recursive algorithm, and the method has a calling relationship with the two algorithms. Therefore, the time complexity level of NRCR is higher and the time is longer. The traditional Dijkstra algorithm is difficult to converge when there are multiple requests due to the simple algorithm results. When the number of requests is small, the RAHR proposed by us divides the algorithm into two steps and needs to connect each node hierarchically, so the computation time is high. Our algorithm has relatively

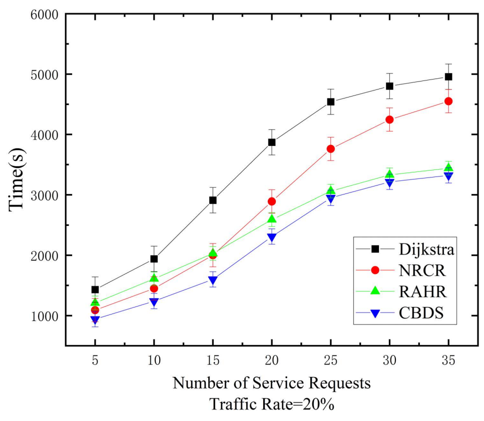


FIGURE 5. Algorithm running time, traffic rate = 20%.

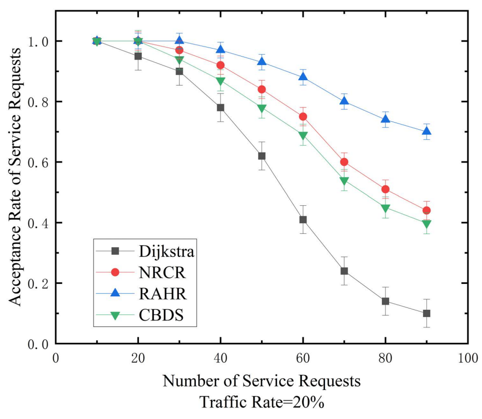


FIGURE 7. Acceptance rate of service requests, traffic rate = 20%.

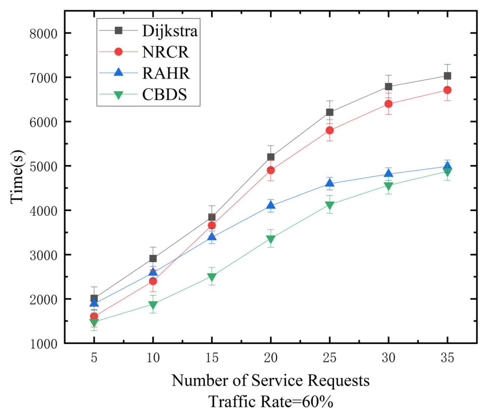


FIGURE 6. Algorithm running time, traffic rate = 60%.

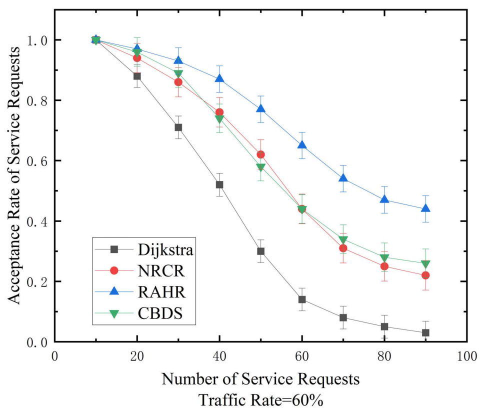


FIGURE 8. Acceptance rate of service requests, traffic rate = 60%.

stable time complexity. Because there are fewer physical nodes in each layer after layering, fewer computations are required between each pair of nodes. Additionally, there is a limited number of VNFs required for a service request. When the number of requests increases, RAHR only needs to allocate a small number of VNFs to complete multiple service requests. When the traffic rate reaches 60%, as shown in Figure 6. Our algorithm is less affected by the traffic rate, and the running time of the algorithm varies less.

The acceptance rate of service requests is shown in Figures 7 and 8. When the traffic rate is low, as the number of service requests increases, the traffic in the network increases sharply, resulting in the fastest decline of Dijkstra algorithm, making it difficult to face complex traffic requests. NRCR improves the minimum reliability of SFC as much as possible, but this not only requires more resource consumption but also only slightly improves the average reliability. Therefore, when a request comes in, NRCR will discard some of the requests in order to improve the lowest reliability. The hierarchical ranking of reliability by CBDS makes this method have a higher reliability guarantee. However, due to the lack of judgment on the reliability of multiple physical nodes, the acceptance rate of requests is not high. RAHR has a higher service request acceptance rate because it considers the joint problem of reliability and cost, and ensures that a highly reliable node and a highly available VNF are selected as much as possible when a request arrives. When traffic rate = 60%, NRCR will drop more requests due to a slower sampling rate and processing speed. Besides, our method allocates physical and virtual resources more efficiently, which enables our algorithm to have higher redundancy in the case of high traffic rates.

We continue to describe the availability of the SFC in terms of the probability of failure of the SFC flow. We define the proportion of SFC flows that can be routed correctly when the same number of requests are received as the failure rate. By Figure 9and Figure 10, it can be seen that Dijkstra lacks traffic scheduling because it only considers the routing problem of the shortest path. Therefore, the probability of failure is the highest. NRCR has a low error rate because it aggregates the backup schemes of multiple SFCs, which maximizes the lowest reliability possible. RAHR has a low failure rate because reliability is considered in the deployment process and a link with higher total reliability is selected. CBDS uses a layered credibility scheme. However, CBDS lacks reliability calculations for multiple physical nodes, which leads to its

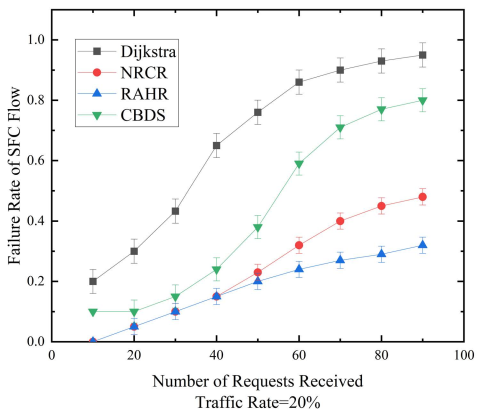


FIGURE 9. Failure rate of SFC flow, traffic rate = 20%.

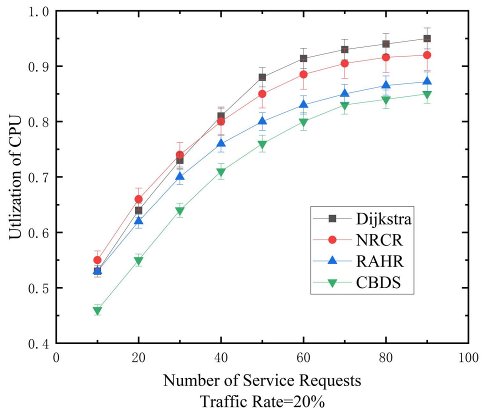


FIGURE 11. Utlization of cpu, traffic rate = 20%.

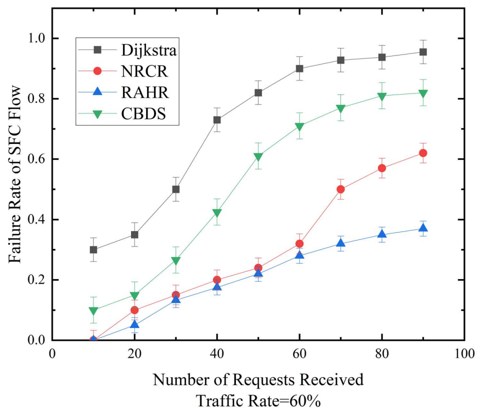


FIGURE 10. Failure rate of SFC flow, traffic rate = 60%.

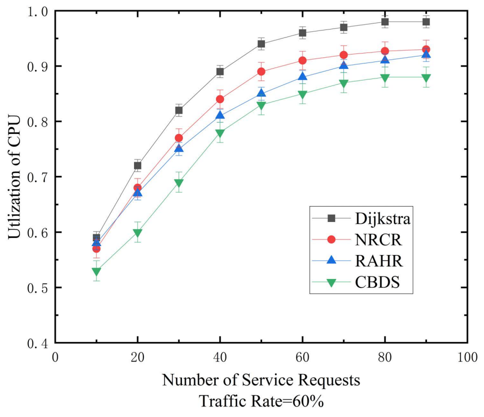


FIGURE 12. Utlization of cpu, traffic rate = 60%.

higher error rate. When the traffic rate is high, it is difficult for CBDS to handle more requests, which leads to more errors in calculating routing paths. The experimental results show that our method has certain advantages in both the acceptance rate of service requests and the failure rate of requested services. Therefore our method has higher reliability.

The computing resource utilization is shown in Figure 11 and Figure 12. When traffic rate = 20%, as the number of service requests increases, the computing resources are saturated the fastest in the routing process of the Dijkstra algorithm. NRCR aggregates too many virtual resources and consumes more computing resources. CBDS ranks the nodes in the network with credibility, and only uses a small number of computing resources each time. Since we divided the physical nodes into layers at the beginning of the architecture design, the calculation of the primary node and the backup node requires our high computing resources. But as service requests increase, more flows can be routed by fewer backup nodes and links, and RAHR will converge faster. When traffic rate = 60%, NRCR needs to aggregate network resources more frequently, which also consumes more computing resources.

The bandwidth resource utilization is shown in Figure 13 and Figure 14. NRCR mainly aggregates the computing resources of the virtual network, which occupies fewer bandwidth resources. In addition, the NRCR stipulates the upper limit of traffic, and the traffic processing rate is exchanged for higher bandwidth. CBDS lacks traffic perception and bandwidth calculation, so it consumes higher bandwidth resources. When traffic rate = 60%, the consumption of bandwidth by CBDS is more obvious. Our approach leverages the global view of the SDN controller as well as our layered approach to quickly find links that meet requirements and save bandwidth. Our approach makes link selection more convenient by rationally deploying VNFs on physical hosts. Therefore, RAHR also performs better at high traffic rates.

The total latency of the SFC is shown in Figure 15 and Figure 16. The total delay includes the computational delay of the physical node and the transmission delay of the link. When traffic rate = 20%, with the increase in requests and nodes, the Dijkstra algorithm is difficult to deal with the situation of multiple request flows. When NRCR processes traffic, it takes a lot of time to filter and combine VNFs multiple times, which leads to higher latency. CBDS pays attention to the reliability of all nodes, so that the selected

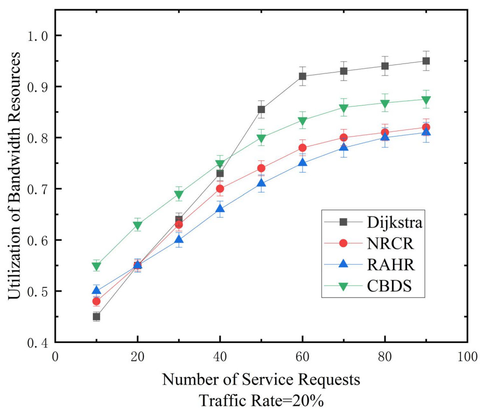


FIGURE 13. Utilization of bandwidth resources, traffic rate = 20%.

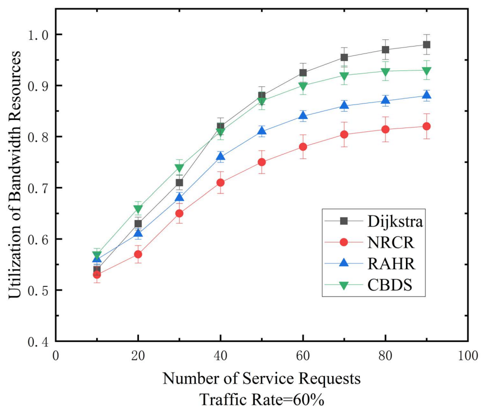


FIGURE 14. Utilization of bandwidth resources, traffic rate = 60%.

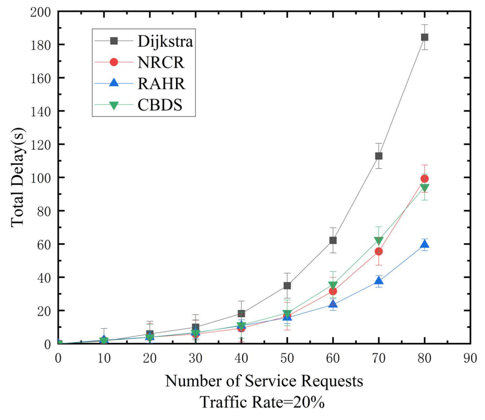


FIGURE 15. Total delay, traffic rate = 20%.

adjacent nodes may have too long distances, which leads to the problem of higher delay. Since the hierarchical network architecture requires multiple steps to process service requests, our method has a high latency in the early stage. But as service requests increase, our method can find physical nodes faster and deploy VNFs to reduce overall latency. When traffic rate = 60%, CBDS requires a longer processing time for each aggregation and alignment. In fact, the SFC-HO

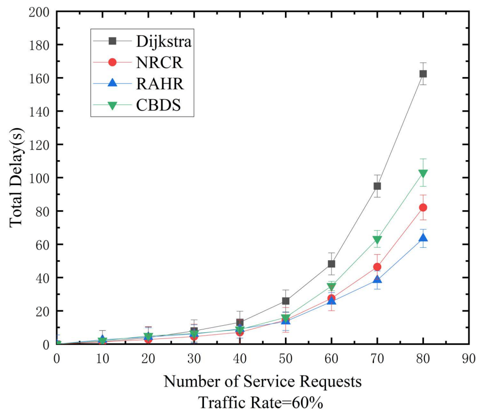


FIGURE 16. Total delay, traffic rate = 60%.

method proposed by us increases the number of hosts connected between functional nodes and the number of links between hosts, which will increase the transmission delay of links to a certain extent. However, our design makes more nodes more reliable, which also reduces the computation time of nodes.

VII. CONCLUSION

In this paper, we discuss the SFC orchestration problem in the context of NFV. We divide the orchestration of SFC into two parts: VNF deployment and path selection. In addition to this, our proposed hierarchical orchestration method divides the physical network and the virtual network hierarchically. We cleverly use a layered approach to allow for a more rational arrangement between servers in the physical network, and layering the virtual network helps manage and filter different types of VNFs. Then, we put forward our goals, and with the resource capacity and delay of nodes and links as constraints, we deploy VNF instances when the dependencies between VNFs are satisfied. Our aim is to maximize the reliability of the SFC while reducing costs. Finally, we propose the RAHR algorithm to solve our problem. The experimental results show that our research can effectively improve the acceptance rate of SFC requests, and has certain advantages in terms of delay and resource utilization. However, we did not consider the situation of multiple network domains in this work, and the current network is mostly composed of multiple domains, and each domain has a certain relationship. In future work, we will consider the problem of multi-domain networks.

REFERENCES

[1] *Network Functions Virtualisation (NFV): Architectural Framework*, ETsI Gs NFV, Eur. Telecommun. Standards Inst., Europe, vol. 2, no. 2, 2013.

[2] M. Mechtri, C. Ghribi, O. Soualah, and D. Zeghlache, "NFV orchestration framework addressing SFC challenges," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 16–23, Jun. 2017.

[3] G. Sun, R. Zhou, J. Sun, H. Yu, and A. V. Vasilakos, "Energy-efficient provisioning for service function chains to support delay-sensitive applications in network function virtualization," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6116–6131, Jul. 2020.

[4] B. Farkiani, B. Bakhshi, and S. A. MirHassani, "A fast near-optimal approach for energy-aware SFC deployment," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 4, pp. 1360–1373, Dec. 2019.

[5] N. Huin, B. Jaumard, and F. Giroire, "Optimal network service chain provisioning," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1320–1333, Jun. 2018.

[6] G. Sun, Z. Chen, H. Yu, X. Du, and M. Guizani, "Online parallelized service function chain orchestration in data center networks," *IEEE Access*, vol. 7, pp. 100147–100161, 2019.

[7] J. Liu, H. Xu, G. Zhao, C. Qian, X. Fan, and L. Huang, "Incremental server deployment for scalable NFV-enabled networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 2361–2370.

[8] X. Shang, Y. Huang, Z. Liu, and Y. Yang, "Reducing the service function chain backup cost over the edge and cloud by a self-adapting scheme," *IEEE Trans. Mobile Comput.*, vol. 21, no. 8, pp. 2994–3008, Aug. 2021.

[9] T. W. Kuo, B. H. Liou, K. C. Lin, and M. J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1562–1576, Aug. 2018.

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

[11] R. Kang, F. He, and E. Oki, "Virtual network function allocation in service function chains using backups with availability schedule," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4294–4310, Dec. 2021.

[12] L. Qu, C. Assi, M. J. Khabbaz, and Y. Ye, "Reliability-aware service function chaining with function decomposition and multipath routing," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 835–848, Jun. 2019.

[13] M. Karimzadeh-Farshbafan, V. Shah-Mansouri, and D. Niyato, "Reliability aware service placement using a Viterbi-based algorithm," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 1, pp. 622–636, Mar. 2019.

[14] M. Pattaranantakul, Q. Song, Y. Tian, L. Wang, Z. Zhang, A. Meddahi, and C. Vorakulpipat, "On achieving trustworthy service function chaining," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 3, pp. 3140–3153, Sep. 2021.

[15] A. Mouaci, E. Gourdin, I. Ljubić, and N. Perrot, "Virtual network functions placement and routing problem: Path formulation," in *Proc. IFIP Netw. Conf. (Networking)*, Jun. 2020, pp. 55–63.

[16] W. Rankothge, F. Le, A. Russo, and J. Lobo, "Optimizing resource allocation for virtualized network functions in a cloud center using genetic algorithms," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 2, pp. 343–356, Jun. 2017.

[17] H. Yu, Z. Zheng, J. Shen, C. Miao, C. Sun, H. Hu, J. Bi, J. Wu, and J. Wang, "Octans: Optimal placement of service function chains in many-core systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 9, pp. 2202–2215, Sep. 2021.

[18] A. Varasteh, M. De Andrade, C. M. Machuca, L. Wosinska, and W. Kellerer, "Power-aware virtual network function placement and routing using an abstraction technique," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.

[19] H. Hawilo, M. Jammal, and A. Shami, "Network function virtualization-aware orchestrator for service function chaining placement in the cloud," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 643–655, Mar. 2019.

[20] M. Jalalitabar, E. Guler, G. Luo, L. Tian, and X. Cao, "Dependence-aware service function chain design and mapping," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[21] S. Sharma, A. Engelmann, A. Jukan, and A. Gumaste, "VNF availability and SFC sizing model for service provider networks," *IEEE Access*, vol. 8, pp. 119768–119784, 2020.

[22] F. Esposito, M. Mushtaq, M. Berno, G. Davoli, D. Borsatti, W. Cerroni, and M. Rossi, "Necklace: An architecture for distributed and robust service function chains with guarantees," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 1, pp. 152–166, Mar. 2020.

[23] B. Farkiani, B. Bakhshi, S. A. MirHassani, T. Wauters, B. Volckaert, and F. De Turck, "Prioritized deployment of dynamic service function chains," *IEEE/ACM Trans. Netw.*, vol. 29, no. 3, pp. 979–993, Jun. 2021.

[24] V. Eramo, F. G. Lavacca, T. Catena, and P. J. P. Salazar, "Application of a long short term memory neural predictor with asymmetric loss function for the resource allocation in NFV network architectures," *Comput. Netw.*, vol. 193, Jul. 2021, Art. no. 108104.

[25] V. Eramo and T. Catena, "Application of an innovative Convolutional/LSTM neural network for computing resource allocation in NFV network architectures," *IEEE Trans. Netw. Service Manage.*, early access, Jan. 11, 2022, doi: 10.1109/TNSM.2022.3142182.

[26] A. Dwaraki and T. Wolf, "Adaptive service-chain routing for virtual network functions in software-defined networks," in *Proc. Workshop Hot Topics Middleboxes Netw. Function Virtualization*, Aug. 2016, pp. 32–37.

[27] H. Moens and F. D. Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *Proc. 10th Int. Conf. Netw. Service Manage. (CNSM) Workshop*, Nov. 2014, pp. 418–423.

[28] Y. Wang, L. Zhang, P. Yu, K. Chen, X. Qiu, L. Meng, M. Kadoch, and M. Cheriet, "Reliability-oriented and resource-efficient service function chain construction and backup," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 1, pp. 240–257, Mar. 2020.

[29] W. Fan, Q. Cui, X. Li, X. Huang, and X. Tao, "On credibility-based service function chain deployment," *IEEE Open J. Comput. Soc.*, vol. 2, pp. 152–163, 2021.

[30] A. Tomassilli, "Towards next generation networks with SDN and NFV," Ph.D. dissertation, Côte d'Azur Univ., Nice, France, 2019.

[31] J. Pei, P. Hong, K. Xue, and D. Li, "Resource aware routing for service function chains in SDN and NFV-enabled network," *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 985–997, Jul. 2018.

[32] M. Nguyen, M. Dolati, and M. Ghaderi, "Deadline-aware SFC orchestration under demand uncertainty," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2275–2290, Dec. 2020.

[33] Y. Chen and J. Wu, "Latency-efficient VNF deployment and path routing for reliable service chain," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 651–661, Jan. 2020.

[34] L. Gao and G. N. Rouskas, "Congestion minimization for service chain routing problems with path length considerations," *IEEE/ACM Trans. Netw.*, vol. 28, no. 6, pp. 2643–2656, 2020.

[35] G. Sun, Z. Xu, H. Yu, X. Chen, V. Chang, and A. V. Vasilakos, "Low-latency and resource-efficient service function chaining orchestration in network function virtualization," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5760–5772, Jul. 2019.

[36] S. Choi, J. Turner, and T. Wolf, "Configuring sessions in programmable networks," *Comput. Netw.*, vol. 41, no. 2, pp. 269–284, Feb. 2003.

[37] M. Jalalitabar, E. Guler, D. Zheng, G. Luo, L. Tian, and X. Cao, "Embedding dependence-aware service function chains," *J. Opt. Commun. Netw.*, vol. 10, no. 8, p. C64, 2018.

[38] Y. Liu, Y. Lu, X. Li, Z. Yao, and D. Zhao, "On dynamic service function chain reconfiguration in IoT networks," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10969–10984, Nov. 2020.

XUANZHE CHEN received the B.S. degree in electronic science and technology from the Civil Aviation University of China, Tianjin, China, in 2020. He is currently pursuing the M.S. degree in communications engineering with Beijing Information Science & Technology University, Beijing, China. His research interests include switching and routing technology and service function chain technology.

JINHE ZHOU received the B.S. and M.S. degrees in radio physics from Wuhan University, Hubei, China, in 1988 and 1991, respectively. He is currently a Professor with the School of Information and Communication Engineering, Beijing Information Science & Technology University. He has been the author of more than 50 articles. He has hosted and participated in several scientific research projects, including the National Key Project of Hi-Tech Research and Development Program of China (973 Program) and the National Natural Science Foundation of China. His research interests include 5G networks, edge computing, game theory, and green information-centric networks. He was a recipient of the Beijing Famous Teacher Award.

SONGLIN WEI received the B.S. degree in communication engineering from the North China Institute of Science and Technology, Hebei, China, in 2021. He is currently pursuing the M.S. degree in communications engineering with Beijing Information Science & Technology University, Beijing, China. His research interests include resource allocation, SFC orchestration, and routing.

• • •