Computer Science Review 38 (2020) 100298

Contents lists available at ScienceDirect

Computer Science Review

journal homepage: www.elsevier.com/locate/cosrev

Review article

A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture

Karamjeet Kaur *, Veenu Mangat, Krishan Kumar

University Institute of Engineering and Technology (UIET), Panjab University, Chandigarh, India

A R T I C L E I N F O

Article history:
Received 24 June 2020
Received in revised form 9 August 2020
Accepted 18 August 2020
Available online xxxx

Keywords:
Network Function Virtualization
Service Function Chaining
Service classifier
Virtual Network Function
Service Function Forwarder
Load balancing of VNFs
VNF placement
Availability

A B S T R A C T

Network Function Virtualization (NFV) has emerged as an innovative network architecture paradigm that uses IT virtualization technology to abstract the network node functions from hardware. The virtualized network services hosted on Virtual Machines (VMs) are called Virtual Network Functions (VNFs). The sequence of multiple VNFs required by network operators to perform traffic steering is called a Service Function Chain (SFC). Software Defined Networking (SDN) is a complementary technology which allows programmatic control of network functions and policy-based resource management. The flexibility of SDN facilitates structuring of SFCs with minimum latency. SFC provisioning using SDN and NFV will enable implementation of next generation 5G networks and make the subscriber/operator relationship more economical and flexible. In this paper, a Systematic Literature Review (SLR) is used to select the high-quality research studies related to dynamic provisioning of SFCs in SDN and NFV. A total of 70 studies available in the literature are analyzed. Thereafter, a layered taxonomy is proposed to classify the literature based on the parameters of optimization approaches for the provisioning of SFCs. Finally, the open research challenges for SFC deployment are identified and discussed. This paper is intended to serve as a ready reference for the research community to develop effective and efficient techniques for SFC provisioning in combined SDN/NFV networks by considering a combination of multiple factors viz. placement of VNFs, load balancing, and availability. It will surely aid Cloud Service Providers (CSPs), Application Service Providers (ASPs), and Internet Service Providers (ISPs) in offering reliable, scalable and high-performance services to their customers.

Contents

* Corresponding author.
E-mail addresses: bhullar1991@gmail.com (K. Kaur), vmangat@pu.ac.in (V. Mangat), k.salujauiet@gmail.com (K. Kumar).

https://doi.org/10.1016/j.cosrev.2020.100298

1. Introduction

In the traditional network architecture, each network device such as Load Balancer (LB), Firewall (FW), Network Address Translation (NAT), Gateway (GW), Deep Packet Inspection (DPI), and Intrusion Detection System (IDS), requires dedicated and expensive hardware for its deployment. The hardware-based traditional deployment phenomenon increases the rigidity and complexity of the network [1]. Due to various technological innovations, the number of users of current networks and the required application services, is rapidly growing each day. Therefore, it is a very tedious and error-prone task for the network operator to store and transfer a large volume of data according to specific requirements using a hardware-based network. Additionally, problems such as need for high investment in capital expenditure (CAPEX) and operating expenditure (OPEX), hinder the ability of network operators to expand their capabilities. Innovations in network technology are required to address the challenges of the traditional network architecture [2].

NFV is a promising paradigm to alleviate the problems of the traditional network approach. NFV converts conventional hardware devices into software-based virtual devices that run on the VM instead of a dedicated hardware appliance [3,4]. The software appliances are called VNFs. The network functionality running on the VM provides a similar service as provided by hardware-based network function. NFV technology offers benefits in terms of reduced capital and operating cost, efficiency, rapid development, and resource sharing among multiple users according to the requirement [5]. Moreover, network functions can be placed anywhere according to the needs rather than using fixed places and dedicated hardware for the deployment, as is the case in a traditional network. NFV is still in its nascent stage with scope for further innovations.

SDN is another novel networking architecture used to separate the control plane from the data plane. The centralized monitoring feature offers many benefits that can help make the network more agile, flexible and innovative [6]. The combination of SDN and NFV enables automated service function provisioning to end users [7,8].

SFC is a popular service paradigm that has been proposed to derive maximum benefits from both NFV and SDN [9]. Service function chaining is defined as an ordered sequence of VNFs and subsequent steering of flows through them to provide end-to-end services. SFC using SDN and NFV, facilitates implementation of 5G network slicing and makes the subscriber/operator relationship more economical and flexible. An example of an SFC request, as shown in Fig. 1, consists of the client (source terminal), set of VNFs (FW, DPI, proxy) in a particular order, and a server (destination terminal). The source and destination are represented using oval shape, and rectangular shape is used to describe different VNFs [10].

1.1. Motivation

The primary motivation behind this work is the evolution of service function chaining from traditional service chaining to dynamic service chaining, which is gaining popularity due to advancement in technologies such as SDN and NFV [11]. Moreover, SFC plays a pivotal role in the implementation of network slicing in 5G networks which are future generation mobile networks [12,13]. NFV is used to virtualize the network functions on a single high-volume server to create a chain of service functions. SDN offers benefits to make the SFC more agile, flexible, and automated. Dynamic service chaining gains maximum benefits from an amalgamation of both SDN and NFV.

Due to the synergy between both NFV and SDN networking technologies, many researchers are working to design and implement provisioning techniques for SFCs in this field [14]. It is also observed that this field is steadily growing in terms of publications. This work is based on using digital libraries such as IEEE (ieeexplore.org), Elsevier (sciencedirect.com), Springer, and ACM for publication analysis. The analysis has been conducted using the query "adaptive service function chaining in SDN and NFV" in these digital libraries from 2013–2020, as illustrated in Fig. 2.

1.2. Related academic research

After a detailed study of the available literature, a few survey papers have been found on adaptive service function chaining in NFV and SDN. The survey paper [9] provides information about SFC standardization efforts and architecture, use cases, and the importance of SFC in NFV to arrive at future research directions. Various optimization approaches for SFC implementation are also discussed in this paper. Another survey paper [15] shows taxonomy of control plane and data plane based SFC solutions and sheds light on the research gaps. The [16] also discusses SFC architecture along with different stages of SFC. Another survey paper [10] provides a closer look at SDN architecture, benefits for the SFC, and also identifies various traffic steering techniques used by SDN based SFC approaches. [17] provides taxonomy for VNF placement solutions in SDN. A close inspection of these existing studies reveals that they fail to provide deep insight into this research field in terms of combining SDN and NFV technologies with the aim of provisioning flexible, dynamic, and adaptive SFCs. An exhaustive coverage of current literature is lacking in the above mentioned [9,10,15–17] papers. Table 1 shows the comparison of our survey with the existing related surveys.

1.3. Contributions

This research article aims to make a distinct contribution by following a systematic methodology to conduct an exhaustive survey in the field of adaptive provisioning of SFCs in SDN/NFV networks. This article endeavors to provide an updated eagle's

Table 1
Comparison of our survey with the existing surveys.

Reference	Systematic Literature Review	SFC architecture	SFC solutions			Taxonomy of SFC provisioning techniques	Research challenges	Research gaps
			Load balancing based	Placement based	Availability based			
[9] 2016	×	✓	×	✓	×	×	✓	×
[15] 2016	×	✓	∗	×	×	×	✓	×
[16] 2018	×	∗	×	∗	×	×	✓	×
[10] 2019	×	∗	×	∗	×	×	✓	×
[17] 2019	×	×	×	✓	×	✓	✓	×
This paper	✓	✓	✓	✓	✓	✓	✓	✓

✓: Full ∗ : Partial ×: None

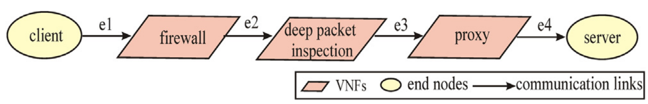

Fig. 1. Service Function Chaining Request.

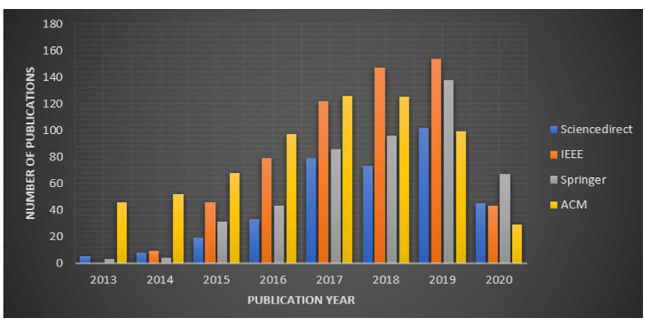

Fig. 2. Number of publications from 2013–2020 in "Service function chaining in NFV and SDN".

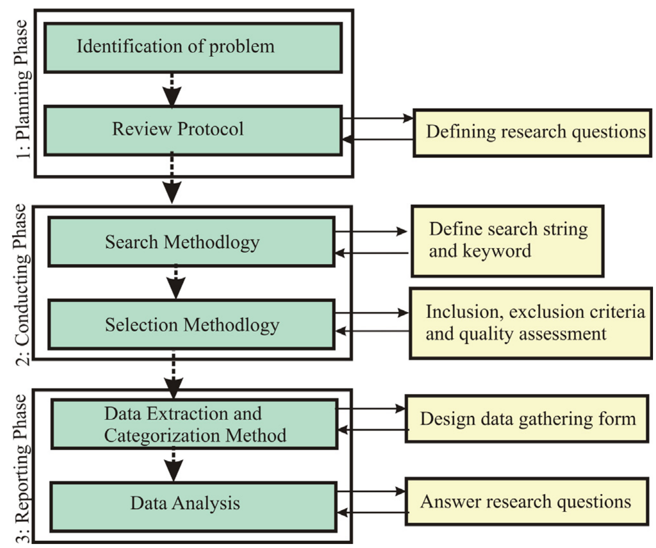

Fig. 3. Systematic Literature Review (SLR) Protocol Process.

eye view of all perspectives of SFCs, based on review of the existing literature. There is an urgent need for systematic scrutiny in this field owing to its vast potential and growing popularity in the telecommunication industry [18]. Techniques for provisioning of SFCs in SDN/NFV networks are highly sought by CSPs, ISPs, ASPs, to enhance their user experience and business profits. In computer science, there are a number of surveys in the field of cloud computing [19] , software engineering [20], and network security [21] that have used SLR. To the best of our knowledge, this survey paper is the first in the field to utilize a survey protocol using a software engineering strategy for conducting a systematic review of the literature. Further, this survey paper proposes a taxonomy for SFC optimization on the basis of load balancing of VNFs, placement of VNFs, and availability, since mobile operators assign chains of service functions to a particular network slice by considering these principles [22,23]. The significant contributions of this paper are as follows:

- Classification of standardized SFC architecture and extended ETSI NFV architecture for the effective development of SFC.
- Layered taxonomy of different optimization approaches such as load balancing of VNFs, placement of VNFs, and availability for effective provisioning of adaptive SFCs.
- Identification of key research challenges in the area of SFC deployment.
- Listing of research gaps in techniques for SFC provisioning for SDN/NFV networks based on analysis of the existing literature.

The complete article is organized in different sections as follows. Section 2 describes the SLR protocol consisting of a series of steps to conduct the literature survey systematically. Section 3 provides a tutorial on SFC provisioning strategies. SFC standardized architecture and extended ETSI NFV architecture are discussed in Section 4. SFC optimization approaches are classified based on load balancing of VNFs, placement of VNFs, and availability, in Section 5. The research challenges are discussed in Section 6. A list of identified research gaps and further research directions are presented in Section 7. Section 8 summarizes and concludes the paper.

2. Systematic literature review protocol

The researchers use a sequence of methodologies for systematic literature evaluation which assist in achieving widespread understanding of the problem at hand. The systematic study offers a constructive method to collect and precisely identify literature related to a particular research problem. Moreover, it enables identification of research gaps that provide directions for further research work. In computer science, there are a number of surveys in the field of cloud computing [19], software engineering [20], and network security [21] that used SLR.

The theoretical view of the SLR protocol, as shown in Fig. 3, consists of a sequence of steps. The first step is to define research questions that will help formulate the search string. A comprehensive literature search is conducted based on the search string to answer the research questions. The series of steps that are needed to perform the systematic research are described in the subsequent sub-sections, along with description of each step.

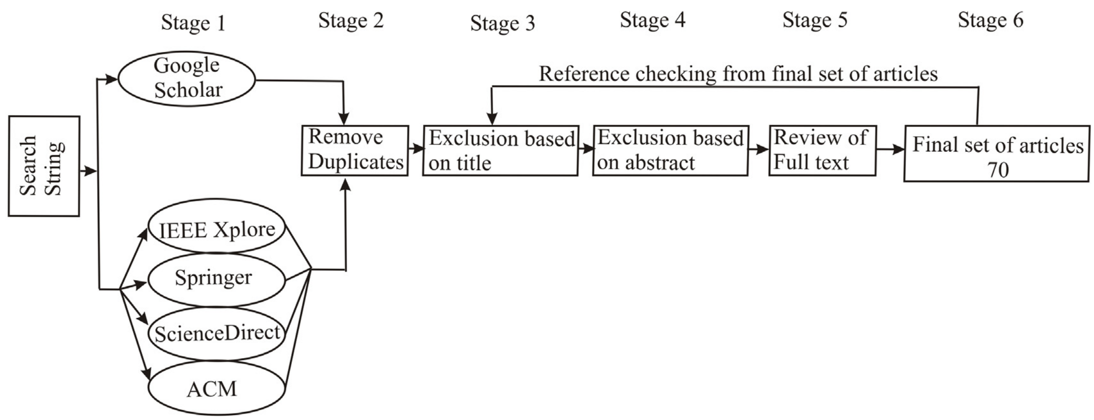

Fig. 4. Systematic Literature Review Process.

2.1. Defining research questions

The construction of research questions is the most crucial step in systematic research. Then, according to these research questions, researchers can scrutinize the searched data and try to find out the answers to the specific set of questions. The following are the four research questions that are addressed in this paper:

RQ1: What are the different strategies used for service function chaining? How do SDN and NFV technologies lend momentum to SFCs? (Section 3)
RQ2: What is the standardized SFC architecture consisting of data plane and control plane components? What is the extended ETSI NFV architecture for SFC? (Section 4)
RQ3: What are the different approaches used by researchers for the effective deployment of dynamic SFCs? What are the strengths and weaknesses of the existing techniques in literature? (Section 5)
RQ4: What are the different open research challenges faced by practitioners and researchers in this field? (Section 6)
RQ5: What are the different research gaps in the existing approaches? (Section 7)

The main emphasis of this article is on answering the above research questions after critically studying all articles related to adaptive service function chaining in NFV and SDN. In RA1, we define the evolution of service function chaining, beginning from traditional SFCs to the current dynamic SFCs. The typical SFC architecture consisting of data plane and control plane components and extended ETSI NFV architecture for SFC is elaborated in RA2. In RA3, we critically analyze the SFC optimization approaches and propose a taxonomy based on load balancing of VNFs, placement of VNFs, and availability, along with their strengths and weaknesses. In RA4, we identify various research challenges that can pave the way and motivate researchers to work in this field. In RA5, we discuss the various research gaps in the existing literature.

2.2. Search methodology

The systematic scrutiny of the existing literature started from finding the appropriate material from the electronic libraries. The search strategy is an important step that can affect the overall performance of conducting the literature survey. In this paper, three phases viz. specific phase, general phase, and reference checking phase, are used to perform the literature research for the period 2015–2020.

- In the specific phase, four digital libraries are considered, namely "IEEEXplore [24]", "Springer [25]", "ScienceDirect [26]", and "ACM Digital Library [27]", to find out the research material corresponding to the research problem.
- In the general phase, the main concentration is on digital search engine "Google Scholar" to find articles related to the research problem.
- In the reference checking phase, references for some of the research articles are analyzed. Then find the final set of papers from recommendations after filtering out the irrelevant articles to our research topic.

The following are some standard search strings used to search within digital libraries with slight alternations in the string.

- Dynamic Service Function Chaining in NFV and SDN
- Dynamic Service Function Chaining in NFV
- Service Function Chaining in Software-Defined NFV architecture
- Adaptive Service Function Chaining

The research articles obtained from all digital libraries using the search query have been narrowed down for further detailed analysis. The filtering criteria is based on title, abstract, and full text, to refine the results for the comprehensive review. Fig. 4 represents different stages of the SLR process [21].

2.3. Study selection criteria

In this section, additional criteria has been adopted to select only relevant work to the problem statement. At this stage, only those studies have been added to the final list that were able to answer the research questions.

In the specific phase, all four digital libraries returned a total of 1481 entries. The general phase fetched a total of 1000 entries from Google scholar. From the specific and general phase, a total of 2481 entries are available in Stage 1. The total number of articles fetched from both the phases were further narrowed down after critical analysis. In Stage 2, 419 entries were withdrawn from the list of total articles obtained from Stage 1 and remaining 2062 entries were passed to the next stage. Then the elimination of items based on title (1465 studies excluded), abstract (419 entries removed), and full text (121 studies eliminated) was done in the subsequent stages. After final Stage 5, a total of 70 studies were extracted as shown in Fig. 5.

The following are the points for inclusion criteria:

- Articles that can provide valuable information regarding the SFC concept in NFV.

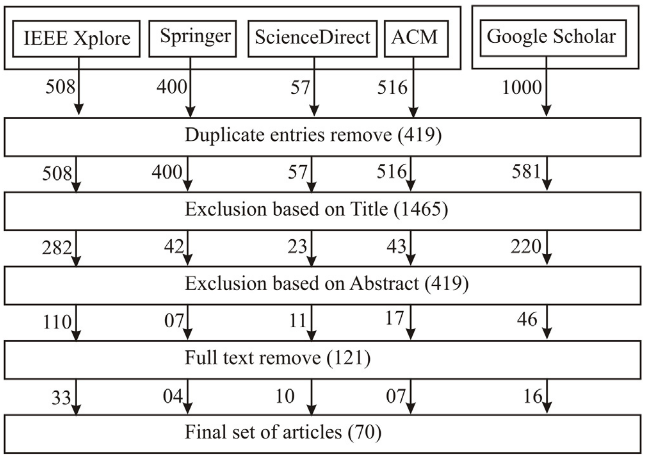

Fig. 5. Total number of studies at each stage of Systematic Literature Review (SLR) Protocol.

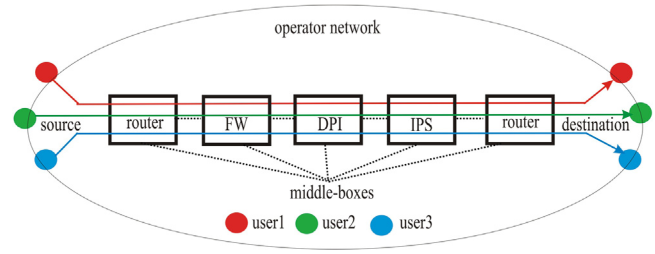

Fig. 6. Static Service Function Chaining.

Table 2
Distribution of the articles according to the type of publication.

Publication type	No.of studies	Percentage
Journal	33	47.14%
Conference	24	33.80%
Symposium	10	14.08%
Workshop	3	4.22%

- Studies that can provide answers to the given research questions.
- Studies related to the deployment of Service Function Chaining with different approaches.
- Studies that can help to differentiate the Static Service Chaining and Dynamic Service Chaining, their respective strengths, and limitations.
- Articles that shed light on SFC architecture along with its components, SDN/NFV based SFC architecture.

The following are the points for exclusion criteria:

- The research work was not able to provide an adequate amount of information.
- The article was not written in the English Language.
- The studies were not related to the research topic.
- The paper was not published between the years 2015–2020.

2.4. Reference checking

The references of final set of articles were analyzed and these articles were passed back to Stage 3 for scrutiny against exclusion criteria defined in subsequent stages. This step reduced the probability of omitting any relevant research article.

2.5. Data extraction

In the last step, remaining high-quality papers were critically reviewed to extract the data from each article for answering the research questions. A pre-designed excel form was filled by analyzing the final set of documents. The spreadsheet consists of fields such as title, objective, criteria for SFC, parameters, methodology, experimentation tool, SFC strategy, number of chains, strengths, weaknesses, and performance metrics. Ultimately, a total of 70 studies were entered with these fields. Table 2 shows the distribution of the articles according to the type of publication i.e. whether it is a journal, conference, symposium or workshop publication.

3. RA1: Service function chaining strategies

3.1. Static service function chaining

In traditional service chaining, network functions are implemented as hardware middle boxes [28], and all are physically connected. FWs, NAT, Web Proxies, LBs are examples of middle boxes used by network operators for security or performance purposes [29]. In the static type of model, every packet, or flow will have to pass through the chain, although some requests need only a subset of these network services. The following are the limitations of this approach [30]:

- Every device should have enough capacity to handle the full traffic, although some have to only pass the traffic without processing it. Therefore, all services are designed with a pessimistic approach [31].
- It is not possible to apply only desired network functions based on specific flow.
- CAPEX and OPEX cost due to purchasing new hardware devices if the existing topology is not able to fulfill it.
- The network devices must be physically connected and manually configured by network operators which may lead to inconsistent configuration.

As shown in Fig. 6, the source wants to send the request to destination and needs only firewall and deep packet inspection network function. But in static service chaining, traffic must pass through the entire network function chain regardless of the need [32].

3.2. Dynamic service function chaining

In dynamic service chaining, SDN and NFV replace traditional middle-boxes with VMs and allow dynamic service chaining [33,34]. In dynamic service chaining, the traffic needs to be steered only through desired network functions according to specific flow requirements [35]. SDN controller can create chains dynamically and forward traffic intelligently to a particular network function based on the label such as VLAN, source MAC address, network service header (NSH). This type of chain is called a software control service chain. The same network function can be used in different function chains. The dynamic service chaining model offers several advantages to operators, such as proper utilization of network and compute resources [36,37]. The following are the advantages of dynamic SFC:

- Service function chaining with SDN/NFV provides greater flexibility for end-to-end service provisioning. The SDN controller will configure the different service chains for a new subscriber by defining a new policy.

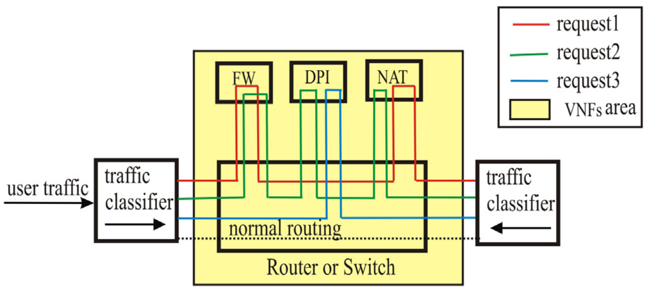

Fig. 7. Dynamic Service Function Chaining.

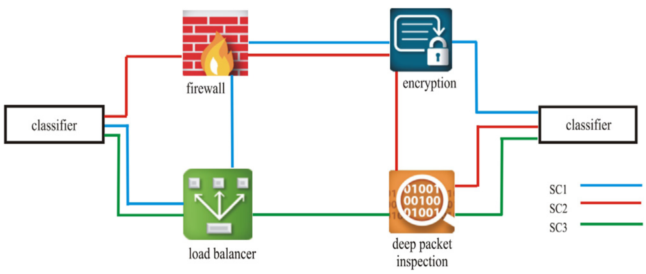

Fig. 8. Different Service Function Chain examples.

- Reduces capital, operational cost as controller steers the traffic to only essential network function and eliminates over-provisioning of the network.
- SDN provides scalable, dynamic, flexible, and automatic service function chaining. Moreover, it gives a better experience for users as providers keep in mind the requirements of users.
- The pessimistic approach is not used to design networks functions. Only those devices will have enough capacity that are used for processing.
- If traffic requirement increases for a particular chain, the ability of only those network functions increases, that are present in that chain.

As shown in Fig. 7, request1 caters to accessing web service from the server and needs only basic firewall and NAT. The SDN controller creates a service function chain for request1: [FW, NAT] in which traffic will pass through the only firewall, and NAT [38]. The request2 caters to accessing critical data from the server, and the controller will create separate SFC of request2: [FW, DPI, and NAT] in which traffic will pass through all the network functions [39]. The request3 will pass through the only DPI using chain: [DPI].

4. RA2: Service function chaining architecture

In service function chaining, the sender sends the traffic into the network that will pass through a chain of VNFs to reach the final destination according to the type of incoming traffic, and user. As shown in Fig. 8, different colors of the line represent different service chains (SCs) consisting of Network Functions (NFs) in a particular order [40]. When traffic enters, the classifier first classifies the traffic pattern into suitable category and then applies the most appropriate SC. Some examples of typical SCs are as follows [41]:

- **SC1:** [LB, FW, Encryption]
- **SC2:** [FW, Encryption, DPI]
- **SC3:** [LB, DPI]

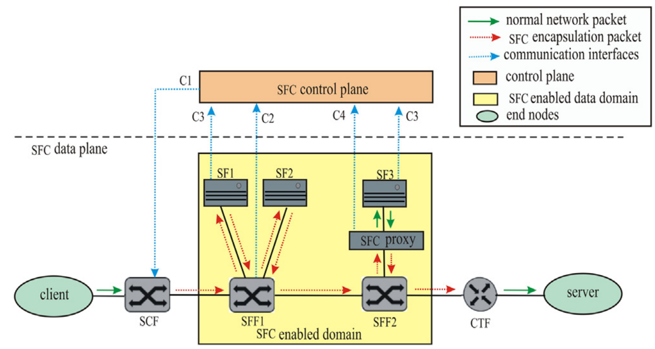

Fig. 9. SFC Data Plane and Control Plane Components.

Table 3
SFC data plane and control plane components.

SFC data plane elements : Responsibility
SCF : Data classification
SFF : Traffic forwarding
SF : Network functionality
SFC-Proxy : SFC encapsulation
SFC : Abstract view of series of SFs
SFP : Actual path of Service Function
SFC Encapsulation : Identification of SFP
SFC enabled domain : Region where SFC is implemented
SFC control plane elements : Responsibility
C1 (Interface between SFC Classifier and SFC control plane) : SFC control plane inserts SFC classification rules into SFC classifier
C2 (Interface between SFF and SFC control plane) : SFC control plane inserts flow rules into SFP forwarding table maintained by SFF
C3 (Interface between SFC aware SFs and SFC control plane) : SFs send their status to SFC control plane
C4 (Interface between SFC Proxy and SFC control plane) : SFC proxy sends information about unaware SF

4.1. Standardized architecture for SFC deployment

The delivery of end-to-end services to the user depends on a series of network functions. The network functions such as FW, IDS, Intrusion Prevention System (IPS), Web Proxy, and Video Optimizer are sequentially processed in a chain called SFC [42]. Internet Engineering Task Force (IETF) [43] has developed SFC architecture, and it defines all the components of SFC [44]. SFC architecture consists of Service Classification Function (SCF), Service Function (SF), Service Function Forwarder (SFF), SFC enabled domain, and SFC proxy, These components communicate with each other with the help of SFC Encapsulation. IETF has defined SDN-based SFC architecture as a two-layered architecture consisting of data plane and control plane components [45].

4.1.1. SFC data plane component

The following elements are present in the SFC data plane component, as shown in Fig. 9. All the standard terms used in the SFC data plane and control plane architecture are shown in Table 3 [46]. .

1. **Service Classifier Function (SCF):** This component is responsible for the classification of data. When a packet enters into the network, SCF classifies and match it with available policies and then chooses the appropriate SFC. It adds SFC encapsulation to define a set of the required service functions [47].
2. **Service Function (SF):** SF is responsible for performing a particular network function. SF is a logical or virtual component used to give specific treatment to a packet. There may be multiple instances of the same network function that can

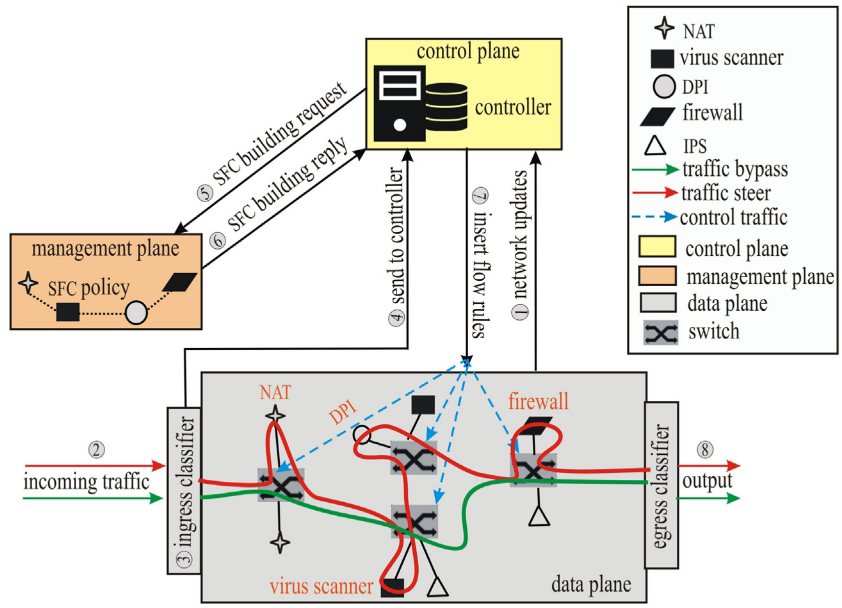

Fig. 10. SDN/NFV based Dynamic Service Function Chaining.

be present in the SFC enabled domain. Network functions can be of two types:If the data sent to SF contains SFC encapsulation (identifier to identify SF), then it is called SF aware. If the data sent to SF does not include SFC encapsulation, then it is called SF unaware.

3. **Service Function Forwarder (SFF):** A number of SFs can be connected using SFF. It is responsible for forwarding the traffic from one SF to another SF or SFF, according to the attached SFC encapsulation. SFF acts as a forwarding device for an incoming packet.
4. **Service Function Chain (SFC):** SFC is an abstract view of SF that is applied to the packet resulting from SCF.
5. **Service Function Path (SFP):** The actual path that consists of a number of SFFs and SFs to steer the traffic, is called SFP.
6. **Service Function Chaining Proxy (SFC-Proxy):** SFC proxy inserts and removes SFC encapsulation for unaware SFs.
7. **Service Function Chaining Encapsulation:** It provides an identifier that is used to forward the packet to particular SFF and SF. It is used for SFP identification.
8. **Service Function Chaining enabled domain:** SFC enabled domain is a network region where SFC is implemented.
9. **Chain Termination Function (CTF):** This is the termination point of the SFC-enabled domain.

4.1.2. SFC control plane component

The following components are present in SFC Control Plane as shown in Fig. 9 [46]. The primary responsibility of the SFC control plane component is the management and controlling of SFC, management of SFs, the mapping of an abstract view of SFC to actual SFP, and inserting rules into SFF components of data plane. This is shown in Table 3. It is responsible for dynamically changing the SFP if any SF or link is overloaded or inactive due to an error [48]. It is also used for administrating and controlling SFC data plane components. The four reference interfaces, namely C1, C2, C3, and C4 are used by the SFC control plane component to communicate with the SFC data plane component [49].

1. **Interface between SFC Classifier and SFC control plane (C1):** This interface is responsible for inserting SFC classification rules into the SFC classifier. Then the classifier only binds the incoming traffic to particular SFP according to that rule. The control plane also updates and deletes the classification rules.
2. **Interface between SFF and SFC control plane (C2):** The SFC control plane inserts flow rules in the SFP forwarding table of SFF using interface C2. Then SFF forwards the traffic according to the rules stored in a table.
3. **Interface between SFC aware SFs and SFC control plane (C3):** This interface is used by SFs to send their status to the SFC control plane. Then the control plane uses this information to update the rules into the SFP forwarding table of SFF.
4. **Interface between SFC Proxy and SFC control plane (C4):** Using this interface SFC control plane can interact with SFC proxy. SFC proxy sends information about the unaware SF to the SFC control plane.

4.2. SDN/nfv architecture for SFC deployment

In modern telecommunication networks, the demand for internet services has increased due to the emergence of novel networking architectures such as the Internet of Things (IoT) [50]. But the static service chaining model is not optimized to provide internet services in terms of reducing CAPEX and OPEX. Furthermore, adding new functionality to existing architecture is hard for network operators [51]. As a result, the dynamic SFC model has become a promising research area. The limitations of the static SFC model can be eliminated by combining key technologies of SDN and NFV. The central controller of SDN architecture has a global view of the whole network topology. Then the controller updates the rules into the flow table of the switch, thus providing centralized control of VNFs [52].

The SDN/NFV based architecture for SFC consists of three types of components, namely — Orchestration Plane, Control Plane, and Data Plane, as shown in Fig. 10. The main responsibility of the Orchestration Plane is to build SFC strategies to control the global network according to different network traffic or user demands [53]. The SDN controller adds flow rules into the OpenFlow table of the switch to orchestrate different VNFs. According to a particular SFC strategy, the SDN controller performs mapping of VNFs and virtual links onto a substrate network and form a SFP [54]. The switches/routers and NFV platforms reside in the data plane and are responsible for the flow of traffic and service processing [55].

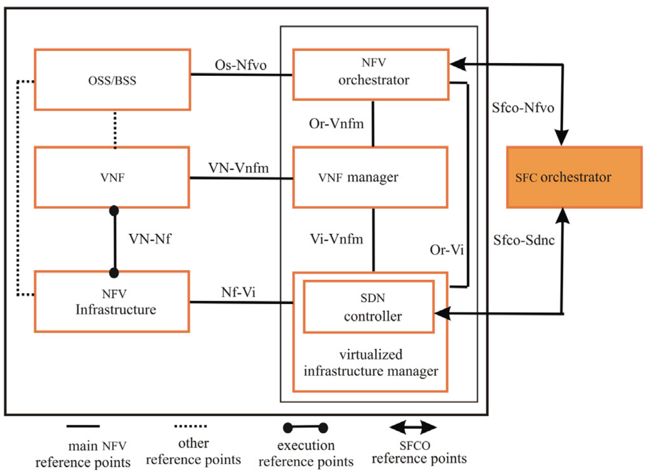

Fig. 11. Extended ETSI NFV Architecture for SFC.

4.3. Extended ETSI NFV architecture for SFC deployment

The new component SFC Orchestrator (SFC control path) is integrated with the existing ETSI NFV framework to provide an effective and flexible SFC deployment in the cloud network as shown in Fig. 11. The NFV Orchestrator (NFVO) component is responsible for managing the operations such as instantiation, deletion, and scalability of network functions using Virtualized Infrastructure Manager (VIM) [56]. The Virtual Network Function Manager (VNFM) is accountable for managing VNF instances (SFs in SFC terminology) [57]. The global view of the whole network is used by the SDN controller to provide flow rules in the flow table of switches. The main component of ETSI NFV architecture is the SFC orchestrator responsible for making VNF Forwarding Graphs (VNFFGs) for network traffic. Consequently, SFC policies are designed for classifiers and SFFs. The new component uses the "Sfco-Nfvo" and "Sfco-Sdnc" interface for communicating with NFVO and SDN controller respectively.

The SFC orchestrator consists of two functional blocks: SFC Manager, and SFC catalogue. The SFC manager is responsible for SFC deployment and its management by communicating with the SDN controller and NFVO. Moreover, it is also responsible for adding, deleting, and updating the operation of SFC and building new SFP on top of the SDN controller. In turn, SFC catalogue is a database that stores all information regarding SFP, SFC, and SFC classifier rules.

The successful SFC deployment depends upon the following activities [58]:

1. **Instantiation and Deletion Operation:** All components present in ETSI NFV architecture such as NFVO, VNFM, and SDN controller are used for the dynamic deployment of SFC. NFVO is responsible for extracting and launching VNFGG and providing different SFPs that defines the specific order of SFs. The VNFM receives requests from NFVO to initiate and delete the VNFs according to a particular SFC request and acknowledge back to NFVO. Then NFVO creates Name Server (NS) records for addition and deletion and sends it to SFC orchestrator [59]. The SFC orchestrator parses the VNFFGs received from NFVO in the NS record. Consequently, the SFC orchestrator creates flow rules of SFP for the SDN controller. Finally, the SDN controller adds flow rules into the flow table of switches. Then it notifies the SFC orchestrator about the successful operation of SFC. This series of steps is depicted in Fig. 12.
2. **Scaling in/out Operation:** For scaling operation, a set of autoscale policies are defined at the VNF level using a threshold value. If the average amount of CPU load, memory utilization increases/decreases from the threshold value, then the monitoring component generates an alarm for scale in/out the operation. The NFVO checks scale in/out of action and then requests VNFM to instantiate or delete the VNFs respectively. VNFM acknowledges back to NFVO after the instantiation/ deletion of VNF [60]. NFVO creates an NS record for the SFC orchestrator to update the SFP and SFC according to scaling operation. The SFC orchestrator calculates the effective SFP and then creates flow rules for the SDN controller. Finally, the SDN controller adds flow rules into the flow table of a switch and notifies SFC orchestrator of the successful scaling operation as shown in Fig. 13.

5. RA3: Service function chaining provisioning techniques

In this section, we have done the in-depth analysis of the final set of 70 research articles that are obtained from Section 2. These research articles are classified into three categories on the basis of provisioning techniques for SFCs because mobile operators assign chains of service functions to a particular network slice by considering these principles three parameters.

1. Load Balancing Aware SFC provisioning (LBA-SFC)
2. PLacement Aware SFC provisioning (PLA-SFC)
3. AVailability Aware SFC provisioning (AVA-SFC)

In the final set of articles, 21 articles deal with LBA-SFC, 28 studies relate to PLA-SFC, and 21 studies provide information regarding AVA-SFC. Fig. 14 presents a state-of-the-art taxonomy that divides the service function chaining solutions into nine categories based on criteria, type of network function, number of chains, SFP strategy, objective, optimization method, network scenario, experimentation tool, dataset, and performance metrics.

5.1. Load balancing aware service function chaining (LBA-SFC)

In this section, the proposed taxonomy considers criteria of VNF load balancing, link load balancing, and both VNF and link balancing, as well as parameters of CPU utilization, memory utilization, disk usage rate, and bandwidth utilization [61], to divide the load balancing solutions in the literature, as shown in Fig. 15 [62]. Table 4 shows some LBA-SFC deployment solutions.

Lee et al. [63] used uniform distribution, and network-aware distribution methods for flow distribution. In uniform distribution, no resource constraint (capacity or link bandwidth) of the service function is considered. The flow is equally distributed among service functions that provide the same service. In network-aware distribution, flow distribution problems are represented as linear programming problems that also consider link bandwidth. The authors proposed optimal flow distribution that can minimize the end-to-end flow latency by considering both capacity and link bandwidth. This model minimizes the end-to-end flow latency. However, they considered only link bandwidth and SF capacity but did not optimize the processing delay and queueing delay at each SFF. In addition, different capacity SF instances may be deployed. Moreover, there is no solution presented by the authors to handle faulty components. Mixed physical and virtual functions may be used for the effectiveness of optimal flow distribution. Medhat et al. [64] provided efficient SFP after selecting appropriate VNFs from multiples by considering the load parameter of VNFs across multiple data centers. They used two algorithms for VNF selection: uni objective and multi objective. The uni objective algorithm considered only load balancing among number of VNFs. The multi objective algorithm

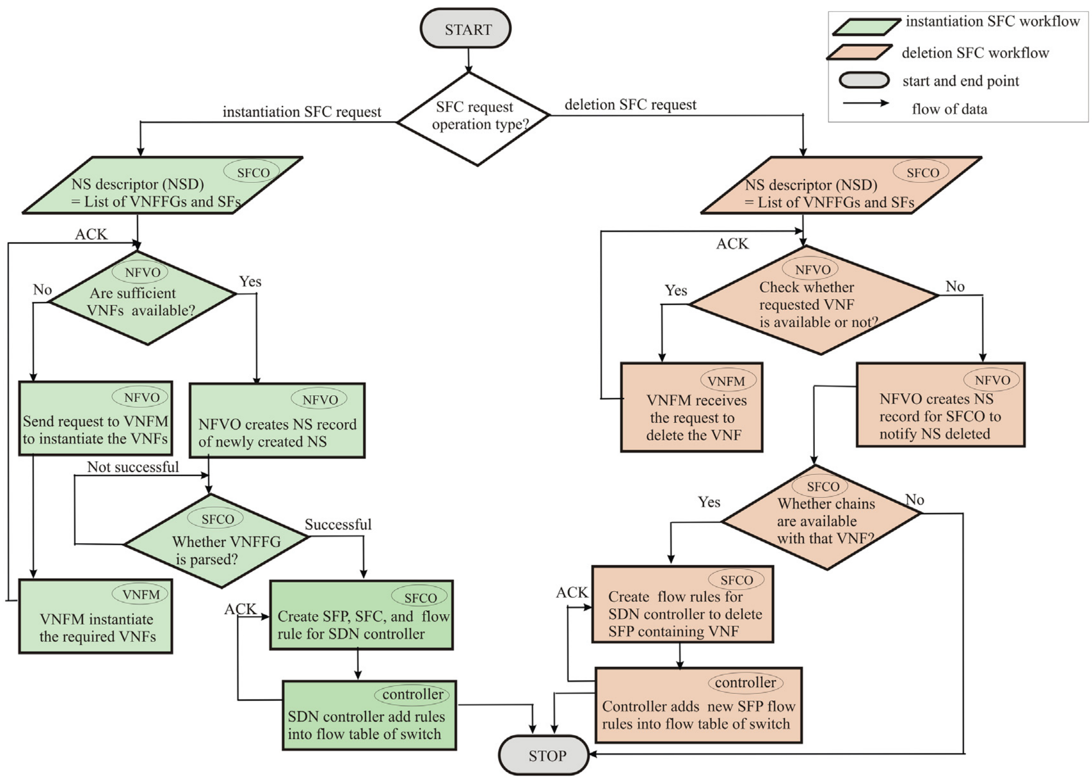

Fig. 12. Instantiation and deletion operation for SFC deployment.

considered multiple objectives such as VNF load, application type, and end-to-end delay for selection. This algorithm optimizes both end-to-end delay and load of service function jointly as compared to other SF selection algorithms. The work can be extended by adding dynamic mechanism in the monitoring of delay and load parameters since they used static value. The proposed algorithm can be tested in the ODL SFC real environment as a future step.

Lin et al. [65] introduced the Hashed Based Load Balancing Scheme named Balanced Hash Tree (BHT) that enables switches to select service functions for chaining without the help of the controller. This is the first work that helps to improve the performance of the system by eliminating the data plane and control plane overhead for selecting the service functions. The SF and Link load balancing should be addressed simultaneously, but the authors considered only SF load balancing. Moreover, the OpenFlow group table is required on switches to implement this approach. Thai et al. [66] developed the 2-phase algorithm named Nearest First and Local–Global Transformation to address both network and server load balancing simultaneously. In the 1st phase of the algorithm, a greedy method is used to create a service chain that considers both server and link load balancing. The next VNF is selected from all VNFs having the lowest latency (network and server) from the current location. In the 2nd phase of the algorithm, searching is used to improve the solution selected in the 1st phase. In this method, a new service chain is built by replacing the existing VNF with another VNF or by swapping the order of VNFs. In this approach, the overloaded server or overloaded path will not be selected to forward the traffic because both server and path load balancing is performed. However, it is a sub-optimal solution because it does not consider all combinations of VNFs and their respective links. Moreover, the proposed work is not implemented in a real environment.

Akhtar et al. [67] used a control-theoretic load balancing approach. The distributed monitoring application running on the VNF host using the RINA framework [68] continuously gets load information. Then it sends this status information to the central controller that uses control theoretic Propositional Integral (PI) algorithm to balance the load. PI algorithm forwards the traffic to the next VNF when the load on the first VNF exceeds its maximum capacity. This is the first work that uses a control theory approach for load balancing of VNFs. They did not handle the scalability and fault tolerance issues in terms of VNFs. Also, they performed only load balancing among VNFs while ignoring link capacity. Thai et al. [69] designed, implemented, and validated load balancing algorithm for SFC named Hash-based Load Balanced Traffic Steering on Soft-switches (HATS). The main idea of this algorithm is that hash-based techniques are implemented on soft-switches for network and server load balancing without the intervention of the control plane. It can perform VNF and network load balancing and also decrease data plane and control plane overhead. Moreover, packets can follow a different path to reduce hash collision. However, the author did not provide any solution to solve the packet reordering problem when the packets belonging to the same flow follow a different path. Furthermore, the proposed approach also suffers from load balancing performance issue due to elephant flow [70].

Hong et al. [71] proposed an improved version of the legacy interface to Network Security Functions (I2NSF). They added a monitoring component in the legacy framework that monitors the load of network security functions (NSF) and sends this information to the controller. Then the controller calculates the load status and other resource usage parameter of each NSF and forwards the traffic to less-loaded NSF. It improves the efficiency of the system by traffic load balancing. But, the authors did not validate their research work in a real environment, such as OpenStack. Moreover, the work has considered only one type of network security function, which is a firewall for packet inspection. Also, packet inspection on a per user basis is not done. Chou et al. [48] developed the VLAN tag and OpenFlow based service function chain (VOFSFC) framework for the SDN environment

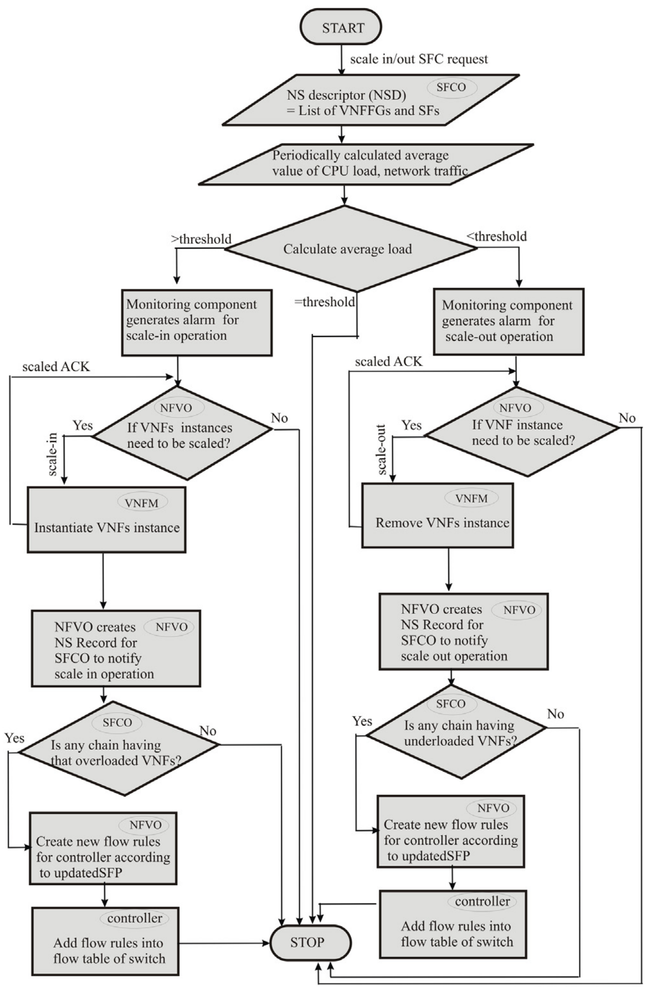

Fig. 13. Scale in/out operation for SFC deployment.

to improve the flexibility of the network and to maximize the network resource utilization. It assigns a unique VLAN ID to each SFP which makes deletion and updation of SFPs easier. The limitation of the work is that it does not consider the load of the virtual network function. The proposed work is not scalable in terms of the addition and deletion of NFs.

Kim et al. [72] formulated dynamic service function chaining by calculating the physical and virtual resource usage by each network function using the reinforcement learning algorithm "Q-learning" which is based on reward and penalty paradigm. The agent performs an action on a particular state on the environment and receives a positive (reward) or negative (penalty) response from the environment. The best-calculated value of the reward (R) is saved for each state. The best reward value is obtained when the load is equally distributed leading to better utilization of resources. They considered both network and VNF resources to select the next node for SFC. But, the scalability is limited in terms of adding and deleting VNF type on different nodes to form a SFC. Moreover, fault tolerance is not included to handle the fault in VNFs. A QoS aware scalable module for SFC orchestration that is capable of dynamically adding new SF instances to handle the overload problem has been developed in [73]. It re-routes the traffic towinin ards a new path if any SF is overloaded and provides QoS aware load balancing SF selection. The proposed framework provides elasticity by adding a QoS aware scalable module as the main module into the SFC orchestrator. But, the proposed framework has been validated with only a small number of chains. Moreover, one chain consists of only a small number of SFs. Scaling in/out should be validated simultaneously. The model can be further extended in terms of multiple SFF and multiple SFs running on each SFF.

Thai et al. [74] introduced subsequent extension of their previous work [69] to solve the hash collision problem. They designed two algorithms based on the previous HATS algorithm. The first, HATS with Flow-Cell Based Multipathing (HATS-Flow-Cell), divides the elephant flows into small flows called flow-cell, and each flow-cell acts as an individual flow and follows a different path. The second, Dynamic Weight Adjustment for HATS (D-HATS) algorithm, updates the weight value (VNFs and path) according to the current load periodically. Overloaded VNFs and

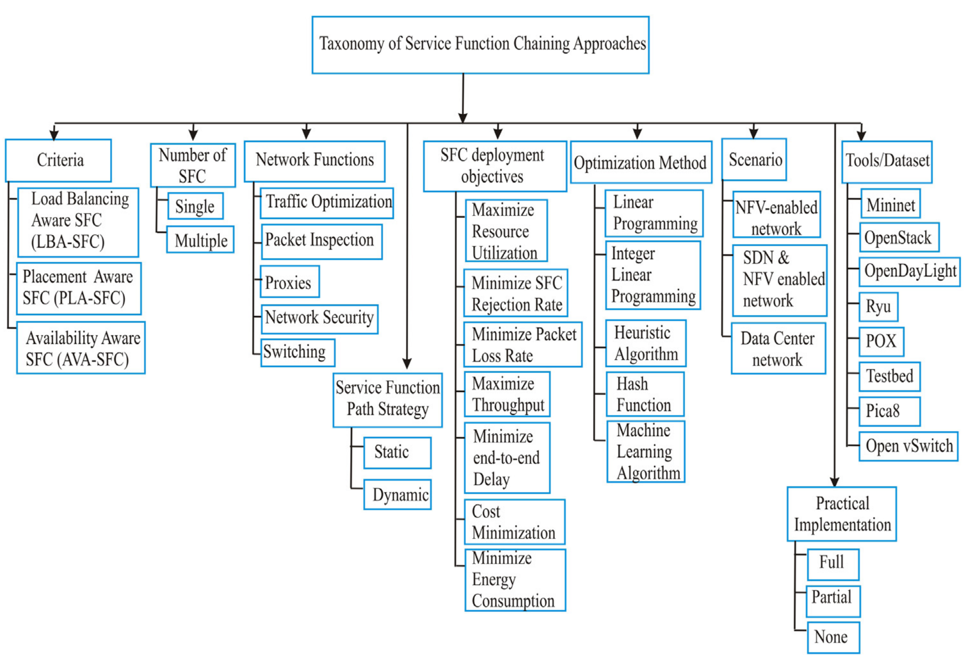

Fig. 14. Taxonomy of SFC approaches.

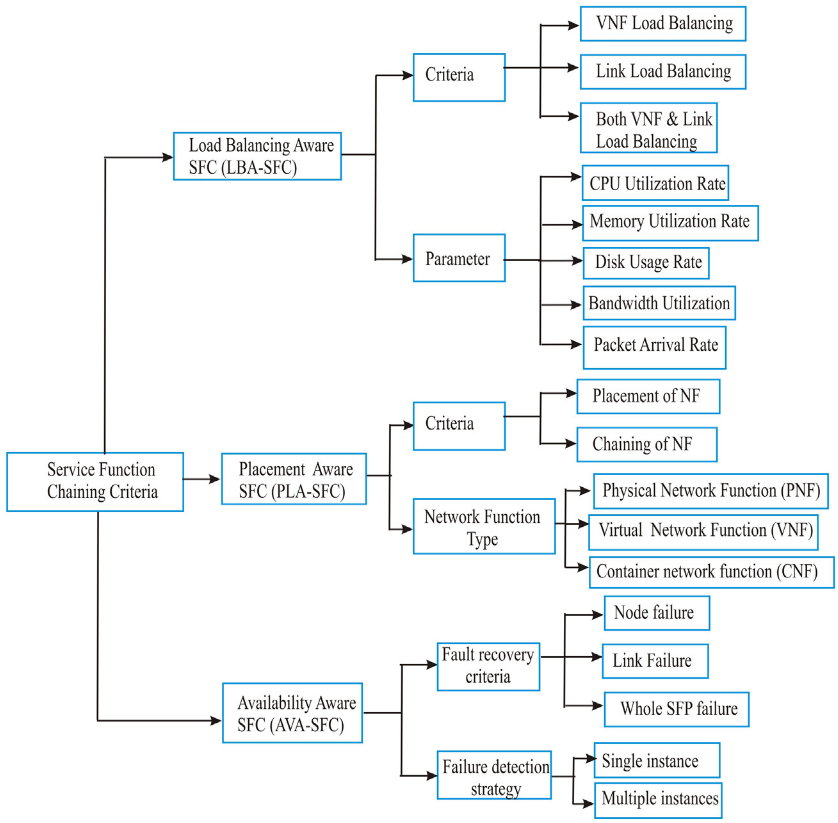

Fig. 15. Criteria wise classification of SFC deployment.

paths have smaller weight values while underloaded VNFs and paths have a higher weight value. These approaches helped to reduce the hash collisions effectiveness. It helps to remove control plane overhead in terms of handling many packet-in messages from the data plane. Also, it helps to improve system performance by reducing flow setup time. However, this paper did not discuss load balancing during the dynamic deployment and scalability of VNFs. The heuristic algorithm named Closed Loop Feedback (CLF) proposed by Sun et al. [75] is based on the consolidation-splitting strategy for network resources. It finds the shortest path for SFC that provides efficient and minimum resource consumption. They also introduced the use of deep learning strategy, Restricted Boltzmann Machine (RBM), to improve the performance of CLF. The limitation is that the authors did not consider the actual link delay between the nodes, instead they randomly assigned the value of delay parameter. Furthermore, the scalability parameter was not considered. Ma et al. [76] developed a framework for the efficient collection of SFPs. In this approach, SFPs having over-loaded service function and service function having price higher than acceptable price, are eliminated. Then the efficiency of remaining SFPs is calculated and the one having greatest efficiency is selected. It provides higher service level agreement (SLA) to users. However, since all nodes are attached to a single switch, so network latency is not optimized. Also, scalability is not present in terms of the addition and deletion of SFs. The resource utilization threshold value is taken as 80%, without suitable explanation.

Hong et al. [77] performed a detailed analysis of a Differentiated Routing Problem for SFC (DRP-SFC) to construct the path for servicing the request by optimally selecting VNFs. Load balancing of VNFs is performed when more than one resource of the same type exists in the network. Resource Aware Routing Algorithm (RA-RA) is proposed to solve the DRP-SFC problem. This is the first work that manages differentiated routing strategies for flow with a particular SFC request. The work can be extended by VNF deployment in the ISP network along with the datacenter. Moreover, SFC failure is not considered by the proposed framework. Also, we can design more efficient differentiated routing algorithm in terms of time efficiency to route the traffic. Heideker et al. [60] used queueing based theory model to define the capacity of VNFs and provide future prediction behavior. The results show that only overloaded functions need to be scaled out to reduce expenditure. The existing studies in the literature used heuristic or AI based techniques for NFV that consider static management. But, they used the queueing model in which future prediction is easy. The work can be extended by adding more number of VNFs or by considering multiple SFCs.

Sun et al. [78] developed a Service Function Chain Deployment Optimization (SFCDO) algorithm based on Breadth-First Search (BFS). It optimizes the delay, resource consumption, and load balancing of VNFs. BFS finds the shortest path from source to destination and then chooses a path with a smaller number of hops to implement SFC. The developed algorithm gave satisfactory results in terms of delay and resource utilization. It considered both resource and link utilization for SFC deployment. Moreover, the load balancing module also improves reliability. However, we can add security network function in the SFC deployed to secure the network. Gu et al. [79] proposed a proactive online learning algorithm named Elastic Virtual Network Function Orchestration (EVNFO) which is a dynamic policy model based on workload prediction. All existing approaches mostly rely on offline policies and ignore the dynamic characteristics of the workload. They implemented Online Instance Provisioning Strategy (OIPS) to dynamically provision VNF instances according to the demand and workload fluctuation. The proactive approach predicting the workload offers significant benefits in terms of cost minimization and service quality.

Table 4
Load Balancing Aware Service Function Chaining (LBA-SFC).

Ref	Aim	Load balancing criteria			Parameters	Number of chains	Service function path strategy	Type of network functions	SFC deployment objective	Optimization method	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		VNF	Link	Both VNF and Link										
[63]	Optimal distribution of traffic among multiple SFs			✓	Node capacity and link bandwidth	Multiple	Static	—-	Minimize flow latency	ILP	NFV network	GT-ITM (topology generator)	Partial	Flow latency, flow dropping rate
[64]	Load balanced and highly available SFC			✓	VNF load, application type, delay	Multiple	Static	—-	Reduce end-to-end delay and improve balanced load	—-	NFV-enabled data center network	MATLAB	Partial	End-to-end delay, SF load
[65]	Server load balanced SFC	✓			Traffic in bytes received from server	—-	Dynamic	—-	Reduce packet-in processing time	Hash algorithm	NFV-enabled network	Ryu controller, OVS, Mininet	Partial	Packet-in processing time, load balancing performance
[66]	Network and Server Load Balancing for SFC			✓	Packet arrival rate	Single	Dynamic	—-	Improve throughput, increase bandwidth utilization	Greedy approach	SDN and NFV-enabled network	Mininet, ODL controller, OVS	Full	Bandwidth utilization
[67]	Control theoretic based load balancing of VNFs	✓			Average CPU load	—-	Static	IDS	Minimize load	Control theory	SDN and NFV-enabled network	GENI testbed, OpenFlow Controller, OVS, RINA framework	Full	CPU load
[69]	Server and link load balanced SFC			✓	—-	Multiple	Dynamic	LF, BS, IF and NAT	Reduce packet-in processing time and system overhead	Hash Algorithm	SDN and NFV-enabled network	ODL controller, OVS, Mininet, Click OS	Full	Flow entries, service chaining time and load balancing performance
[71]	Load balancing of network functions based on monitoring	✓			System state, CPU and memory usage rate, disk usage rate, remaining disk space	—-	Dynamic	FW	Improve packet throughput and resource utilization	—-	SDN and NFV-enabled network	ODL controller, Mininet	Full	Throughput
[48]	Load balancing of SFC and routing		✓		Traffic in bytes	Multiple	Static	FW, IDS, NAT, Web Optimizer, Video Optimizer, DPI, Monitoring	Improve resource utilization	—-	SDN and NFV-enabled network	Pica8 OpenFlow Switch, Ryu Controller, Hypervisor	Full	Traffic statistics
[72]	Learning resource usage using reinforcement learning for SFC			✓	CPU, Memory, and network usage	—-	Dynamic	—-	Equal load distribution	Reinforcement learning	SDN and NFV-enabled network	ODL Controller, OpenStack	Partial	CPU utilization, arrival time for file
[73]	QoS aware load balancing for scalable SFC	✓			CPU load, network usage	Multiple	Static	FW, HHE, PC	Minimize cost and delay	—-	SDN and NFV-enabled network	OpenStack, ODL controller	Full	Traffic load, packet loss rate
[74]	Load balanced SFC			✓	—-	Multiple	Dynamic	LF, BS, IF and NAT	Reduce packet-in processing time and system overhead	Hash function	SDN and NFV-enabled network	ODL Controller, OVS, Mininet	Full	Flow entries, service chaining time and load balancing performance
[75]	Cost-efficient SFC			✓	—-	Multiple	Dynamic	—-	Optimal resource utilization and cost minimization	Heuristic algorithm	NFV-enabled network	US Network Topology	Partial	Communication delay, acceptance ratio, deployment time cost, resource utilization

(*continued on next page*)

Table 4 (*continued*).

Ref	Aim	Load balancing criteria			Parameters	Number of chains	Service function path strategy	Type of network functions	SFC deployment objective	Optimization method	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		VNF	Link	Both VNF and Link										
[76]	Load and cost-effective service function selection for SFC	✓			CPU utilization rate, memory utilization rate, network throughput, unit price	Multiple	Static	—-	Maximize resource utilization, minimize SFC rejection rate and packet loss rate	—-	SDN and NFV network	Mininet, ODL controller	Full	Utilization rate, standard deviation, rejection rate for SFC request, packet loss rate
[77]	Resource aware routing for SFC			✓	Link Bandwidth, end-to end delay, CPU utilization, flow entries of switch	Multiple	Static	—-	Load balancing and minimize resource consumption	—-	SDN and NFV network	MATLAB, CORONET CONUS Topology	Partial	Throughput, hop count, load balancing, and acceptance rate
[60]	Load balanced SFC	✓			—-	Multiple	Static	FW, NAT	Reduce complexity	Queueing theory model	SDN and NFV network	Pica 8 OpenFlow switch, XEN hypervisor	Partial	Throughput, latency, service rate
[78]	Less delay and resource efficient SFC			✓	Resource consumption, end-to-end delay	Single	Static	—-	Maximize throughput and proper resource utilization	Heuristic algorithm	SDN and NFV network	OpenStack, Chinese Network Topology	Full	End-to-end-delay, bandwidth consumption, load rate of node
[79]	Workload prediction based dynamic SFC	✓			Aggregated flow rate	Multiple	Dynamic	FW, Encryption, IDS, NAT	Cost minimization	Mathematical model	NFV network	Geant Topology	Partial	CPU utilization, throughput, operational expenditure
[80]	Network load balance based parallelized SFC			✓	Computing resources, bandwidth capacity	Multiple	Dynamic	—-	Minimize queueing delay and improve load balancing and acceptance ratio	Heuristic Algorithm	SDN and NFV-enabled DC network	US wide Network topology	Partial	Acceptance ratio, link utilization, queueing delay, mapping cost
[81]	Delay guarantee resource allocation for SFC			✓	Computing resources, bandwidth capacity	Multiple	Dynamic	FW, Encryption, IDS, NAT	Minimize end-to-end delay	—-	NFV-enabled network	Geant Topology	Partial	Delay, operational expenditure, acceptance ratio, workload of VNF
[82]	State aware SFC		✓		Bandwidth, delay, packet loss rate, link cost	Single	Dynamic	—-	Maximize throughput Minimize path cost and end-to-end delay	Machine Learning Approach	SDN and NFV-enabled network	Containernet emulator, Topology generator, ONOS, OpenvSwitch	Full	No of successful SFC requests
[83]	Microservice based traffic steering	✓			Threshold capacity	Multiple	Static	Firewall, VPN gateway	—-	—-	NFV enabled 5G core network	Kubernetes platform (K8), docker	Partial	Deployment time, network latency, throughput

Sun et al. [80] proposed a parallel VNF chain for mice flow [84] to alleviate the problems incurred by elephant flow [70] such as long queueing delay and flow completion time. They designed an online parallelized heuristic algorithm (ONP-SFO) based on the worst-fit strategy. First, they divided user requests into several sub-flows and also replicated the same number of sub-SFCs. Then each sub-flow is forwarded into one of the parallelized sub-SFC according to worst-fit strategy. The results show that this algorithm outperforms in terms of improving load balancing, acceptance ratio, and reducing delay. This work can reduce the hash collisions discussed by Thai et al. [69] with the deployment of parallel VNF-SFCs. The flow splitting operation, according to latency-aware, throughput-aware, and security-aware demands, was not considered in the experiment. To optimize resource allocation while guaranteeing the service delay, Gu et al. [81] proposed LGOS (Layered Graph-based SFC orchestration Scheme). In this approach, resource cost and its corresponding delay are labeled as a weight in a graph, which helps to find out the shortest path. The proposed scheme able to reduce operational expenditure, end-to-end delay, and improve the acceptance ratio. This study proposes batch processing algorithm to process a number of SFC requests in a single batch.

Jeong et al. [82] proposed a machine learning approach to measure the network performance quickly and then select the optimal path for the new SFC request. The existing mathematical model cannot provide an optimal solution for scalable problems. The proposed machine learning approach considers network parameters and finds a suboptimal solution. The work can be extended by adding VNF node characteristics (CPU usage) along with link parameters into existing model. The proposed model can be tested in a real environment such as OpenStack. Dab et al. [83], authors discussed the 5G traffic steering in micro services based NFV network. Then they proposed weighted round robin load balancing algorithm to balance the traffic.

5.1.1. Discussion

In Load Balancing Aware SFC provisioning (LBA-SFC), our proposed taxonomy considers criteria of VNF load balancing, link load balancing, and combined VNF plus link load balancing to segregate the existing load balancing solutions in literature. In the final set of articles, 21 articles deal with LBA-SFC. This category of methods considers parameters such as CPU utilization, memory utilization, disk usage rate, and bandwidth utilization, in order to balance the load among different VNFs and links with the objective of improving resource utilization and minimizing latency. After critically reviewing all the research articles, it can be summarized that the design and implementation of load balancing based SFC solutions is still an open research area that needs to be addressed properly. Most of the researchers have examined the VNF load balancing, but link based load balancing is not explored in depth in the literature. Further, the existing methods can be improved by adding fault tolerance mechanisms to make the network robust against failures at VNF or link level. Additionally, the load balancing solutions should be scalable in terms of adding or deleting more network functions

5.2. Placement aware service function chaining (PLA-SFC)

This section presents a discussion of the proposed taxonomy considering criteria of NF placement and NF chaining, and NF type as Physical NF (PNF), VNF, and Container NF (CNF), to divide the VNF placement solutions in literature. This is shown in Fig. 15. Placement aware SFC deployment solutions are discussed in this section and presented in concise form in (Table 5).

Ghribi et al. [85] designed Dynamic Programming (DP) algorithm for fast, efficient, and combined placement and chaining of VNF within polynomial time. The DP optimization approach divides large problems into sub parts, then obtains the solution of large problem by combining the solutions of sub-problems. The computed value is stored in the form of a matrix. The diagonal elements represent the cost of placing the VNF(i) on a particular node j. The non-diagonal elements represent the cost of steering the traffic from VNF(i) placed on node j to the next VNF(i+1) placed on node k. This study focuses on link constraint because it does not force VNF embedding on the same physical node. Moreover, VNF placement and chaining are jointly addressed by the authors. Xiong et al. [86], the authors formulated the Configurable Network Service Chaining (CNSC) as an integer linear programming problem and modeled Service Path

Computation Algorithm (SPCM) to compute optimal SFC by considering the node and link constraints with segment routing. The proposed solution provides flexible service to users by combining both SDN and NFV framework. However, the proposed model can be extended to adopt dynamical construction of the service chain.

Mechtri et al. [87] proposed an extended version of Umeyama's eigen decomposition approach [88] for joint VNF placement and chaining. They also offered multi-tenancy service model and chaining of heterogeneous resources such as VNFs and PNFs. The existing Umeyama's approach required graphs having the same size to perform matching and performed only link mapping. Heterogeneous resources are considered in this study although existing literature is limited to either only PNFs or VNFs. Moreover, this is the first work that considered eigen decomposition approach for VNF placement and chaining and supported multi-tenancy. Furthermore, the proposed approach is faster and scalable as compared to other algorithms. The eigen decomposition approach can be extended by adding multiple objectives and constraints to generalize the model. Wang et al. [89] studied the resource allocation problem (NFV-RA) that consists of three phases: VNF chain composition (VNF-CC), VNF forwarding graph embedding (VNF-FGE), and VNF scheduling (VNF-SCH). Then they developed jointly optimized resource allocation algorithm (JoraNFV) based on mixed integer linear programming to solve the resource allocation problem. They presented two models in JoraNFV: one hop scheduling (OneHop-SCH) and multipath greedy (MPG) model. The study makes it possible to dynamically allocate resources (CPU, memory) to the requested chain more flexibly with the help of SDN controller. However, the work can be improved to reduce the execution time by launching multiple VNFs into a single docker or a Linux container server rather than implementing each VNF on different servers. Heuristic model can be implemented to reduce execution time for the deployment of VNFs.

Hsieh et al. [90] proposed a framework in which network functions are implemented on the SDN switches to minimize the traffic inside the network. The results show that reduction of up to 2/3 of original traffic in a network is achieved using this method. Moreover, performance is also increased as it reduces packet latency. However, network functions implemented on SDN switches is not powerful as traditional network functions. Kim et al. [91] developed an efficient SFC algorithm in order to fulfill the user's QoS requirements while minimizing the energy consumption. In this algorithm, first shortest path is calculated from source to destination that can satisfy the latency constraint and then VNFs are deployed on VMs along that path. It also offers reconfiguration of SFC when energy consumption exceeds a given threshold value. However, only limited types of VNFs are considered which make it impractical to use in real environment. Soualah et al. [92] developed an Energy Efficient Tree-search based Chain placement Algorithm (EE-TCA) for efficient VNF placement and chaining. It helps to reduce the power

Table 5
Placement Aware Service Function Chaining (PLA-SFC).

Ref	Aim	Placement criteria			Number of chains	Service function path strategy	Type of network functions	SFC deployment objective	Optimization method	Scala-bility	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		Place-ment	Chain-ing	Type of SFs										
[85]	Fast and efficient placement and traffic steering	✓	✓	VNFs	–-	Dynamic	–-	Maximize acceptance rate, resource utilization, and revenue	Dynamic Programming algorithm	Yes	Network Provider	GT-ITM topology generator	Partial	Acceptance rate, resource utilization, execution time, and revenue
[86]	Optimal and flexible SFC	✓	✓	VNFs	–-	Static	–-	Maximum network utilization	Heuristic algorithm	No	SDN and NFV enabled network	ONOS controller, NetFPGA-10G, ClickOS, GT-ITM	Full	Request acceptance ratio, resource utilization ratio
[87]	Joint placement and steering flows	✓	✓	VNFs and PNFs	Multiple	Static	–-	Maximize revenue and acceptance rate	Eigen decomposition method, heuristic method	Yes	NFV, Cloud and DC network	GT-ITM topology generator	Full	Convergence time, acceptance rate and revenue
[89]	Optimal placement and service function chaining	✓	✓	VNFs	Multiple	Static	–-	Minimize execution time, maximize network performance	MILP based heuristic algorithm	No	SDN, NFV, and DC network	–-	Partial	Execution time, cost performance
[90]	Traffic steering through VNFs running on OpenFlow switches	✓		VNFs	Multiple	Static	FW, IDS. Proxy	Minimize network traffic	–-	No	SDN and NFV enabled network	OpenFlow switches, SDN controller	None	Number of hops
[91]	Energy aware SFC	✓	✓	PNFs	Multiple	Static	NAT, Proxy, Firewall, router, LB, Flow monitor, IDS, DPI	Maximize QoS, Minimize energy consumption	Genetic Algorithm (GA)	No	NFV and Cloud DC network	OMNET ++ 4.6	Full	Energy consumption, average service latency
[92]	Cost and Energy efficient Service function placement	✓	✓	PNFs and VNFs	Multiple	Dynamic	FW, Proxy, IDS, NAT	Minimize cost and power consumption	Decision Tree Model	Yes	SDN, NFV and Cloud network	GT-ITM, OpenStack, ODL controller	Full	Rejection rate, resource usage, power consumption, execution time
[93]	Scalable SFC provisioning	✓	✓	VNFs	Multiple	Static	Transcoder, compressor	Cost minimization	ILP, Heuristic Algorithm	Yes	NFV-enabled network	OpenStack	Full	Execution time, acceptance ratio, average delay, overall cost
[58]	Cost efficient and scalable solution for VNF-CPP problem	✓	✓	VNFs	Multiple	Dynamic	DPI, Proxy, Transcoder	Cost and time minimization for function placement	Heuristic Algorithm	Yes	NFV-enabled network	–-	Partial	Execution time, Acceptance rate, average cost, and revenue
[94]	NSH based flexible SFC		✓	VNFs	Multiple	Static	DPI, TC, IC	increase flexibility in SFC	–-	No	SDN and NFV network	ONOS controller, OpenFlow Switch, KVM	Partial	Throughput
[95]	Delay optimized SFC		✓	VNFs	Multiple	Static	FW, NAT, TS	Minimize latency, maximize throughput	–-	No	SDN and NFV network	Controller, Berkeley Extensible Software Switch (BESS)	Partial	Latency, throughput

(*continued on next page*)

Table 5 (*continued*).

Ref	Aim	Placement criteria			Number of chains	Service function path strategy	Type of network functions	SFC deployment objective	Optimization method	Scala-bility	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		Place-ment	Chain-ing	Type of SFs										
[96]	Practical implementation of SFC		✓	VNFs	Single	Static	Router, FW, DPI	Minimize overhead	—-	No	SDN and NFV network	OpenStack, OpenvSwitch, OpenDayLight, Ixia (traffic generation)	Partial	—-
[97]	Energy efficient SFC provisioning	✓	✓	VNFs	Multiple	Static	VOC, IDPS, FW, WOC, NAT, TM	Minimize energy consumption	ILP, heuristic method, column generation	No	SDN and NFV network	—-	Full	Execution time, energy consumption, saved energy, link load, delay, accuracy
[98]	Optimal SFC placement and routing	✓	✓	VNFs	Multiple	Static	NAT, Firewall	Minimize server and link consumption, improve network utilization and balance load	ILP, greedy heuristic algorithm	No	SDN and NFV-enabled DC network	NetFPGA-10G, OpenDayLight, sFlow, MATLAB, GT-ITM	Full	Request acceptance rate, node load balancing, path load balancing, running time
[99]	Bandwidth and delay optimized SFC	✓		VNFs	Multiple	Static	NAT, FW, TM, WOC, IDPS, VOC	Minimize end-to-end latency, network resource	Heuristic algorithm based on Column Generation	Yes	NFV network	—-	Full	Bandwidth usage, Execution time
[100]	Dependence-aware SFC	✓		VNFs	Multiple	Static	FW, LB, Encryption	Minimize bandwidth consumption	—-	No	NFV network	US Backbone network Topology	Partial	Average bandwidth consumption
[40]	Bandwidth utilized SFC	✓	✓	VNFs	Multiple	Static	NAT, TS, Encryption, Firewall, IDS, QOS	Minimize resource consumption	ILP based scheme	No	NFV network	—-	Partial	Network resource consumption
[101]	Service function placement with minimum resource cost	✓		VNFs	Single	Dynamic	—-	Minimize network resource cost	Heuristic Method	No	SDN and NFV network	SDN controller	Partial	Path computation time, NFP computation time, feasibility delay, Migration time
[102]	Delay optimized SFC		✓	VNFs	Single	Static	—-	Minimize delay, and resource consumption	Heuristic Method	No	NFV network	—-	None	Average delay, resource usage
[103]	Service function chaining in heterogeneous domains		✓	VNFs and CNFs	Single	Static	VM based forwarder and CN based forwarder	Reduce overhead	—-	No	SDN, NFV, and Kubernetes	OpenStack, K8 (Kubernetes), SONATA GUI (to access both OpenStack and K8)	Partial	ICMP Ping request
[104]	Resource utilized VNF embedding and scheduling	✓	✓	VNFs	Multiple	Static	—-	Increase resource utilization and QoS demand	Heuristic Algorithm	No	SDN and NFV network	—-	Partial	Acceptance ratio, QoS, resource utilization, execution time
[105]	Composition, Placement, and assignment of SFC	✓	✓	VNFs	Multiple	Dynamic	—-	Increase VNF utilization and reduce link consumption	Heuristic Algorithm	No	Cloud DC network	Fat-tree topology	Partial	VNF instance utilization, Link utilization
[106]	Time efficient VNF placement	✓	✓	VNFs	Multiple	Static	Firewall, IDS, Encryption, and VPN	Improve time efficiency and minimize cost	—-	No	NFV network	fat-tree topology	Partial	Acceptance ratio, server utilization, bandwidth usage

(*continued on next page*)

Table 5 (*continued*).

Ref	Aim	Placement criteria			Number of chains	Service function path strategy	Type of network functions	SFC deployment objective	Optimization method	Scala-bility	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		Place-ment	Chain-ing	Type of SFs										
[107]	VNF placement in batch mode	✓	✓	VNFs and PNFs	Multiple	Static	—-	Minimize cost and energy consumption	Heuristic algorithm	Yes	NFV network	GT-ITM, GEANT Topology	Partial	Rejection rate, acceptance ratio, execution time, resource usage, and power consumption
[108]	Energy-efficient and traffic-aware SFC	✓	✓	VNFs	Multiple	Static	—-	Minimize energy consumption	ILP approach, heuristic algorithm	No	Multi-domain cloud network	GT-ITM, GEANT topology, SDN controller, OpenFlow switch	Full	Average server and link power consumption, average power consumption, average response time
[109]	Optimal SFC scheduling	✓	✓	VNFs	Multiple	Dynamic	—-	Improve resource utilization and decrease scaling frequency	ILP approach	Yes	DC network	SDN controller, BCube Topology	Partial	Throughput, resource utilization, scaling frequency
[110]	Efficient mapping of SFC	✓		VNFs	Multiple	Static	—-	Improve resource efficiency	Deep Q-learning based heuristic algorithm	No	Mobile Network	MATLAB	Partial	Average accepted requests, time-cost value
[111]	Cluster based placement and chaining	✓	✓	VNFs	Multiple	Static	—-	Minimize energy consumption, delay, and bandwidth usage	Heuristic algorithm	No	—-	—-	Partial	End-to-end delay, run time, rejection rate

consumption of network devices that are involved in available SFP. Efficient use of energy is accomplished by consolidating the physical resources and sharing VNFs among multiple tenants. A decision tree is incrementally built based on Monte-Carlo search tree that offers scalability features for large scale networks. However, they did not consider the fault tolerance of link or VNFs in terms of measuring its effect on power consumption.

Yi et al. [93] used reactive and proactive strategies to solve scalable SFC provisioning problem (s2FCp2). The reactive scheme (SFC-RS) aims to fulfill the scalable demand without changing SFP. On the other hand, proactive approach (SFC-PS) prepared resources beforehand to serve incoming requests. The results show that SFC-RS is better for large scale networks but SFC-PS is better for small networks. The proposed solution not only handles the SFC request, but also considers scale-in (SI) and scale-out (SO) procedure for dynamic addition and deletion of VNFs. However, we can increase or decrease the VNF resources such as bandwidth for the deployed SFC. Khebbache et al. [58], authors proposed Perfect-2 matching, matrix optimization and multi-stage graph method to solve the VNF Chain Placement Problem (VNF-CPP) in polynomial time for small and large scale graphs. Then they compared these three algorithms on the basis of execution time, acceptance rate, and average cost to find the optimal solution. Their proposed heuristic solution can scale according to the demands. But, the authors did not consider the replications of VNF to solve VNF-CPP problem in polynomial time. Moreover, the remaining resources are not utilized for the next request.

Davoli et al. [94] proposed a framework that uses OpenFlow protocol to control the network nodes and successfully adds dynamicity and programmability to the network. SDN technology is used to implement dynamic SFC using OpenFlow protocol. The proposed work used mixed proactive and reactive approach to install flow rules into the flow table of switch. The work can be enhanced by replacing it with a reactive approach to increase flexibility. Zhang et al. [95] proposed an algorithm ParaBox for parallel packet processing across multiple network functions to reduce SFC latency for delay sensitive applications. The first component, order dependency function, is responsible for checking the dependency of data and can be processed in parallel. The second component, mirror function, distributes packets to the parallel network functions. In the end, a merge function combines the packets after processing to produce the output. This is the first study that worked on parallel packet processing to optimize the SFC latency and maximize the throughput. However, it is very difficult to determine which VNFs can be parallelized. Moreover, the calculation of VNF order dependency, mirror and merge functions, should be optimized in terms of processing time.

Another approach for design, implementation, and validation of SFCs in real environment is given in [96]. In this approach, all the valid SFPs are stored in the database and then service classifier classifies the incoming traffic into the appropriate service category. If traffic does not match with the traffic classes, then default path is followed. This paper offers a new OpenFlow based solution in which there is no need of static encapsulation and de-encapsulation of the packet. However, we can add a scalability module to dynamically add or delete VNFs. Also, they did not consider the load of network functions or link during provisioning of SFC. Huin et al. [97] studied how NFV can be coupled with SDN to improve energy efficiency of the network for SFC provisioning by adapting the utilization of network resources. They proposed ILP formulation to solve Energy Efficient SFC Provisioning (EE-SFCP) problem for small networks. They proposed ILP based heuristic algorithm named GREENCHAIN to solve EE-SFCP problem for large networks. Lastly, they presented two variants of the column generation model: CG-cut and CG-cut+ based on ON–OFF approach of power model. They showed how virtualization technology can reduce the energy consumption and improve efficiency of the network.

Liu et al. [98] formulated SFC placement problem as an integer linear programming problem and proposed heuristic solution to solve placement problem. They modeled modified version of two step mapping algorithm by adding new greedy module, sorting module and k-Dijkstra strategy. They formed Greedy node mapping with k-shortest path link algorithm (G-kSP). In this paper both parameters (node and link) are considered for the placement. In future work, a failure-resilience module can be added to handle failure of VNF nodes or links. Huin et al. [99] proposed exact scalable mathematical model with decomposition scheme for the placement of VNFs. The proposed framework avoids data passing through unnecessary devices in order to minimize the bandwidth consumption and end-to-end latency. The exact number and location of VNFs are also considered as factors for arriving at an optimal solution. They offered a routing solution for SFC request of up to 50 nodes within a few minutes. However, they considered a static scenario in which all service functions and requests are known in advance.

Jalalitabar et al. [100] developed an efficient algorithm, namely Dependence aware SFC Group Mapping (D_SFC_GM) for dependence aware SFC design and mapping while considering the resource demand, and function dependency of the user request. The proposed algorithm is able to jointly optimize the design and mapping for the construction of SFC chain. The algorithm called Dependence aware SFC design and Adaptive mapping (D_SFC_AM) designed by authors did not take the benefits of joint optimization. The existing algorithm can be extended by mapping multiple VNFs node onto the same substrate node. Gupta et al. [40] developed a mathematical model to service chain the VNFs while minimizing the network resource consumption (bandwidth consumption) to fulfill the user needs. This analysis will help network operators to choose the appropriate service chain strategy, according to their OPEX and CAPEX budget. This is the first work that practically considers the location and number of NFV capable nodes and their impact on resource consumption. However, they did not consider the "NFV ALL" strategy in the experiment. Furthermore, SDN is not considered to centrally manage all service chain strategies.

Gadre et al. [101] formulated that Network Function Placement (NFP) problem deals with placing network functions at the right location so that SFC requests can be satisfied. Existing static NFP solutions that use divide-and-conquer (DCA) cannot handle real time requests and are also slow. A heuristic version of the algorithm (DCA-H) is proposed, which is more agile and faster. To handle dynamic requests, they proposed an algorithm that is a combination of divide-and-conquer and modified version of Dijkstra. They proposed a centralized and SDN enabled solution. Existing studies are either working in the routing of flow or optimal placement of network functions individually, but the focus of this work is on both the paradigms. However, the scalability in terms of adding new VNF is not optimized. Furthermore, the fault tolerance module is not considered to handle the failure of a link or VNFs. Cheng et al. [102] proposed heuristic algorithm to implement delay sensitive SFCs. It is able to reduce complexity by minimizing both delay and resource consumption simultaneously. The algorithm can be improved by varying flow rates dynamically. Moreover, queueing delay can also be considered for better optimization.

Kouchaksaraei et al. [103] proposed a Pishahang framework that chains the VM based services and CN (container) based services implemented over Kubernetes and OpenStack environment. Overall cost and latency is reduced if the combination of VMs and CNs is used to deploy network functions rather than using only VMs. But, more complex VNFs such as DPI, NAT, IDS have not been considered in the experiment. Cao et al. [104] formulated, mixed integer linear programming (MILP) based heuristic model for dynamic VNF embedding and scheduling while improving resource utilization. The proposed algorithm is divided into two stages: first greedy strategy is used for VNF embedding and scheduling. Then to satisfy the delay demand, re-embedding and re-scheduling is carried out. The method offers dynamic VNF embedding and scheduling that can change according to the service demands. The work can be extended by considering the link propagation delay for better QoS. Wang et al. [105] developed heuristic algorithm named Jcap to solve SFC-CPA (Composition, Placement, Assignment) problem. They jointly addressed the SFC composition, placement, and scheduling of traffic steering in a data center environment. The proposed method is able to service the online arrival demands. The proposed heuristic algorithm can be extended for general topology rather than data center topology and considering the end-to-end service across a wide area network. Qi et al. [106] modeled multi-stage graph and greedy algorithm for VNF placement. The proposed algorithms provides a time efficient solution by defining accessible scope for limiting the search space. The authors proposed time efficient solution for VNF placement without degradation of quality.

Soualah et al. [107] modeled batch R-ILP approach to reduce the limitations of ILP. They proposed online and batch algorithm that process the requests in a batch mode to reduce the complexity and improve performance. The online approach takes into account the sharing of VNFs by multiple SFCs. The batch algorithm services the requests jointly according to batch window rather than sequential online ILP. The algorithm outperforms other algorithms such as Monte Carlo Tree Search (MCTS) in all performance parameters. Only a limited number of candidates are searched for placement and chaining, and this improves overall performance. The R-ILP approach can be extended to RBR-ILP by re-submitting the batch requests that are not satisfied in the previous time window. Sun et al. [108] proposed a heuristic algorithm named Energy Efficient SFC Orchestration across Multi Domain (EE-SFCO-MD) as an optimal solution to handle online SFC requests. This is the only work that has implemented energy efficient orchestration to handle online SFC requests in a multi-domain network.

A Four Stage Adaptive Scheduling Mechanism (FSASM) to meet both dynamic flows and network performance requirements has been suggested in [109]. NP hardness of FSASM is proved and then a minimum weight path selection scheme (MEPS) is discussed. The scheme takes polynomial time for SFC scheduling while reducing extra scaling and cost. The proposed novel scheduling algorithm can handle dynamic SFC requests without degradation of network performance and ensures lower network cost. However, FSASM should be validated in real NFV-based scenarios. Li et al. [110] showed that efficient mapping of multiple SFCs into the substrate network is very challenging where SFC requests contain multiple resource demands with different priorities like in a 5G network. To alleviate the complexity of ILP solution, they proposed two heuristic algorithms based on Deep Q-learning for SFC mapping into substrate network. They used a dummy resource pool to evaluate resource efficiency of different strategies. They considered the multiple priority constraint SFC and balance the node and link utilization. The learning process is expensive for agent and the learning speed can be improved by simultaneously updating of Q-values. A cluster based placement and chaining approach that divides the substrate network into set of on-demand clusters to optimize energy consumption, delay and bandwidth usage is developed in [111]. When a cluster is set up, then heuristic based solution is used to place and chain the dynamic request. This is the first work that used the cluster based approach for placement and chaining. In the future, further investigation of cluster techniques can be done to improve the quality of the clusters and to allow dynamically changing the number of clusters according to network state.

5.2.1. Discussion

In PLacement Aware SFC provisioning (PLA-SFC), the criteria used for developing the proposed taxonomy are — NF placement, chaining of NFs, or combined placement and chaining. NF placement deals with deploying VNFs on VMs and chaining defines how traffic steers through these VNFs. The existing VNF placement solutions have been segregated by considering the type of network function such as Physical NF (PNF), Virtual NF (VNF), and Container NF (CNF). In the final set of articles, 28 articles deal with PLA-SFC. A careful study of these techniques shows that the scalability issue has not been addressed during implementation of dynamic SFC provisioning strategies. Most of the existing literature considered a static scenario in which all service functions and requests are known in advance. Further, researchers can include energy efficiency as a parameter in the orchestration module to handle SFC requests. Additionally, employing CNFs based on Kubernetes technology for development of scalable SFC solutions, instead of VNFs, may yield more benefits because containers can direct access the hardware with little virtualization overhead.

5.3. Availability aware service function chaining (AVA-SFC)

This section presents the proposed taxonomy considering fault recovery criteria of node failure, link failure, and whole SFP failure, and fault detection strategy of single instance and multiple instances, to categorize resilient-aware solutions in literature. This division is shown in Fig. 15. Table 6 represents some Availability Aware SFC deployment solutions at a glance.

Lee et al. [112] presented a self recovery scheme to handle failure of any service function in order to minimize the signaling delay among control plane and data plane. Self recovery scheme temporally handles the failure by assigning the workload of failure SF to another SF in the data plane without intervention of the control plane. The proposed work helps to improve quality by reducing the delay between data and control plane. However, the performance of the proposed work is not validated. Optimal algorithm is not defined for the selection of remote SFF from multiple SFFs. Jeon et al. [113] elaborated the mechanism to improve the availability of the network by allowing VNFs to be outsourced by the third party. The proposed work offers significant advantage in terms of scalability due to outsourced VNFs by the third party.

Herker et al. [114] presented the model for the high availability of SFC by creating different backup strategies. They selected Data Center (DC) topology that offers better performance in terms of cost per throughput for a given availability level of SFC and provides resilient embedding. The configuration, management, and control traffic is not mentioned and they considered only switch and server failure. They assumed link availability is 100% for simplicity. Medhat et al. [56] presented resilient orchestration feature of SFC that is able to create a new chain at any time as well as re-route the traffic into new SFP if any fault appears. This feature helps to keep the high availability and reliability of service. The strength of this paper is that authors have validated their proposed work in real testbed. The proposed solution can be enhanced to handle a more complex scenario with the simultaneous failure of VNF and link.

To detect any type of failure in SFC, the author of the paper [115] proposed alarm based monitoring mechanism for high availability of SFC. The main strength of proposed mechanism is to provide SFC reliability. There is no auto scaling module for adding new VNF when monitoring component generates alarm after failure detection. The existing framework can be enhanced by adding alarm based link failure detection. Bijwe et al. [116] proposed SFC embedding method that can ensure end-to-end reliability of service chain and provide highly available service. To provide end-to-end reliability, fault avoidance approaches are

Table 6
Availability Aware Service Function Chaining (AVA-SFC).

Ref	Aim	Fault recovery criteria			Failure detection of multiple instances	Recovered SFP after failure	Type of network functions	Optimization method	SFC deployment objective	Scalability	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		Node failure	Link failure	Whole SFP										
[112]	Self-recovery scheme to handle fault in SFC	✓			No	Static	—	Optimal algorithm	Minimize delay between data and control plane	No	—	—	None	—
[113]	VNF outsourcing by third party	✓			—	Static	—	—	Increase availability	Yes	Service-provider network, cloud network	—	None	—
[114]	High Availability of service function			✓	Yes	Static	—	—	End-to-end high availability	No	NFV based Data Center (DC) Network	—	Partial	Cost per throughput, availability
[56]	Resilient orchestration or VNF fault recovery in SFC	✓			No	Static	Firewall	—	Service delivery high availability	No	Cloud Network	Openstack and ODL	Partial	Recovery time
[115]	Fault management using alarm-based monitoring for SFC	✓			Yes	Static	—	—	Increase availability	No	Cloud Network	Openstack and ODL	None	—
[116]	End-to-end reliable SFC	✓			—	—	LB, FW, CPE, DPI	Heuristic Algorithm	Improve availability of service and reduce cost	—	NFV-enabled network	—	Partial	Average service downtime
[117]	Service failure detection and localization	✓	✓		Yes	Static	—	Heuristic Algorithm	Minimize detection cost	No	Service provider network	Fat tree topology	Partial	Recall, False-positive, and forwarding time
[118]	Distributed failover mechanism	✓			Yes	Static	—	—	Reduce failure time	No	NFV-enabled network	—	Partial	Average failure time
[119]	Link failure recovery		✓		Yes	Static	DPI, NAT	Monte-Carlo Search Tree	Minimize service interruptions due to link outages, provide reliable path	No	Cloud network	—	Partial	Rejection rate, acceptance revenue, penalties
[120]	High available, and load balanced SFC	✓			No	Static	FW1, DPI1, FW2, DPI2	—	Minimize overloading and failure of SF	No	SDN enabled service provider network	ODL controller	Partial	Failure recovery
[121]	Provide Resiliency for SFC	✓			Yes	Static	—	Heuristic Algorithm	Minimize resource redundancy and Cost	No	SDN and NFV-enabled operator network	SDN controller, java-based simulator	Partial	Time to locate alternative server, MTTF
[122]	Reliable SFC			✓	—	Static	—	Heuristic Algorithm	Reduce CAPEX and OPEX, improve reliability	No	Telecomm Operator Network	GT-ITM tool	Partial	Resource consumption, deployment time, and block ratio
[123]	Highly reliable SFC	✓	✓		Yes	—	—	—	Minimize resource consumption and cost and improve QOS	No	Data Center Network	Fat tree topology	Partial	Average bandwidth, unavailability rate
[123]	Combine VNF-path backup			✓	Yes	Static	Load balancer, Video encoder	—	Minimum resource consumption and improve availability	No	NFV-enabled network	—	Partial	Maximum availability, resource consumption, total data rate, acceptance ratio, cost

(*continued on next page*)

Table 6 (*continued*).

Ref	Aim	Fault recovery criteria			Failure detection of multiple instances	Recovered SFP after failure	Type of network functions	Optimization method	SFC deployment objective	Scalability	Scenario	Tools/Dataset	Practical implementation	Performance metrics
		Node failure	Link failure	Whole SFP										
[124]	Cost effective and highly available SFC	✓			Yes	Static	––	Heuristic Algorithm	Minimum resource consumption	No	NFV-enabled network	––	Partial	Network resource consumption, loss ratio, CPU resource utilization
[125]	resource aware backup for VNFs	✓			Yes	Static	––	Heuristic Algorithm	Minimum resource consumption	No	NFV-enabled network	––	Partial	Resource consumption, convergence rate
[126]	Reliable SFC	✓			Yes	Static	––	––	Improve resource utilization, and reduce cost	No	NFV-enabled network	––	Partial	Cost, node and link usage rate, service acceptance rate
[127]	Reliability and delay aware SFC	✓	✓		Yes	Static	––	Heuristic Algorithm	Minimize delay and resource consumption	No	SDN and NFV-enabled network	––	Partial	Resource utilization, average SFC reliability, average SFC delay
[128]	Cost efficient reliable SFC	✓			Yes	Static	––	––	Minimize cost and VNF redundancy, increase SLA	No	NFV enabled network	python simulator	Partial	Redundancy deployment cost, number of admitted services, average SFC reliability
[129]	Reliable SFC	✓			Yes	Static	––	Q learning algorithm	Minimize cost, and delay	No	SDN and NFV-enabled network	MATLAB, GT-ITM tool	Full	Request acceptance ratio, resource consumption, running time
[130]	Reliable SFC with minimum cost	✓		✓	Yes	Static	Firewall, NAT, content delivery network, IDS, DPI	Heuristic Algorithm	Resource cost minimization	No	NFV-enabled Carrier Network	––	Partial	Execution time, total network resources cost, acceptance ratio

used to continue system service by embedding highly used VNF on the highly reliable resource. Authors showed that proposed method is able to reduce total downtime as compared to traditional method. Fault avoidance method improved end-to-end reliability by increasing priority of mostly used VNFs. However, they considered limited SFC embedding examples to validate their work.

Zhang et al. [117] developed a mechanism for service failure detection and failure localization in the existing SFC with minimum cost. In failure detection, total failure detection path is calculated. The failure localization analyzes the location of failure. The authors elaborated the existing work by using the unidirectional and bidirectional network. The proposed method is able to reduce detection cost and improve false-positive rate of failure localization. The work can be extended by adding a scalability module to diagnose failure in a dynamic environment. Suh et al. [118] proposed a distributed failover mechanism, in which failover agent is placed with each SF instance to recover from failure of SF. It is shown that distributed failover mechanism is able to reduce failover time as compared to centralized mechanism. The approach is also able to reduce control-data plane communication by recovering from SF failure without intervention of the centralized SFC controller. The failover mechanism can be implemented using open source software and tested using real testbed.

Soualah et al. [119] developed R-SFC-MCTS algorithm based on Monte-Carlo Search Tree for SFC placement and chaining while minimizing the effects of physical link failure. They used preventive and reactive approaches to handle the link failure. This is the first work that used MCTS for SFC placement and chaining in which only partial tree construction is needed to take any decision. The proposed solution is dynamic in nature and reduces complexity. However, they considered only physical link failure while SFC placement and chaining. The work can be enhanced by considering node failure, scaling, and VNF migration. Lee et al. [120], authors developed Overloading and Failure Management (OFM) that consists of Overloading Module (OM) and Failure Module (FM) for SFCs to handle overloads and failure of SFs. The OM module periodically monitors the SF to check the load of SF and FM detects failure of SF using alarm. This is the only work that recovered from the failure while considering the load of service function. The developed module can be extended for NFV environment and VNFM can be used to create a backup SF instance for overloaded and failed SF.

Karra, and Sivalingam [121] designed two algorithms to provide resilient services with minimum resource redundancy. The algorithms either migrate the network service from failed to functional server or improve robustness of the failed component of SFC. The combination of proactive (critical SFC) and reactive (non critical) approaches are used to handle failure instead of only proactive approach used in prior literature studies. The limitation is that only server failure is considered, and link failure is ignored by the authors. Sun et al. [122] developed Ensure Reliability Cost Saving (ER_CS) algorithm to reduce capital and operational expenditure of TSPs by reducing the reliability of SFC deployment. They further proposed ER_CS_ADJ algorithm that can minimize the resource consumption. They offer a tradeoff between resource consumption and reliability for effective use of resources.

To improve reliability of the system, weights are assigned to the important VNFs so that only important VNFs are redundant in paper [123]. Reliability is offered while reducing the need of resource consumption. The proposed work can be extended by considering the dynamic network configuration. Moreover, we can assign weight to VNF by considering other parameters apart from bandwidth. Wang et al. [131] developed availability model by considering both VNF and hardware failure. Then they proposed Joint Path-VNF (JPV) model to consider the VNF-Path backup jointly. They also designed the priority algorithm for SFC that minimizes the resource consumption and guarantees availability. They considered a hardware failure along with VNF failure that is missing in prior in literature. Moreover, they also solved the VNF and path back up jointly. The work can be enhanced to optimize the service chain composition for a large scale network.

Torkamandi et al. [124] developed Availability-aware clustered SFC Embedding (AV-SFC) to handle multiple failures at the same time and reduce the need of multiple resources for backup by using Share Protection Cluster (SPC). Zhang et al. [125] developed Resource Aware Backup Allocation (RABA-CODE) for VNF while minimizing the backup resource consumption and maximizing the availability. They also considered heterogeneous resource requirements for each VNF. To reduce the overhead they also proposed greedy RABA. They offered an efficient solution in terms of backup resource consumption by considering the heterogeneous resource requirements for each VNF. But, they assumed that the VNF is static and has independent availability, which is not the case in practical situations.

Tang et al. [126] developed Reliability-Aware Service Chain Mapping (RSCM) algorithm. They used Improved Breadth-First Search (IBFS) to get cost-effective routing path. Then they also proposed a P*Q replication model that can improve the reliability with QoS requirements. The algorithm proposed by the authors is able to provide a cost efficient solution that ensures low latency and high reliability. They also considered load balancing of the backup node to avoid excessive load in a network. Load balancing module should consider the load of both VNF and link. The proposed algorithm can be improved to handle dynamic service chain requests. Qu et al. [127] formulated the optimal VNF placement and traffic routing problem as a mixed integer linear programming problem. They developed a heuristic algorithm that consists of VNF decomposition strategy for backup along with delay-aware multipath routing to increase reliability of the network. They handled optimal VNF placement and traffic routing jointly that was previously missing in the literature. Further research direction can be to decrease the delay that gets introduced due to function decomposition. Dinh et al. [128] proposed Cost-aware Criticality based priority Index (CCI) parameter to calculate the priority level of VNF for redundancy. Based on CCI, they developed cost efficient VNF Redundancy Allocation (CCI-RA) algorithm. They developed redundancies for only appropriate VNFs based on a priority index that is able to reduce costs. Moreover, they considered the VNF redundancy deployment cost, and priority index, for the evaluation of VNF redundancy. They did not consider the optimal method for primary VNF deployment. Liu et al. [129] also formulated the reliability-aware service chaining problem as mixed integer linear programming problem and then developed Joint Protection (JP) redundancy model and backup redundancy model with cost, delay, and reliability constraints. The proposed algorithm provided cost efficient and reliable solution jointly without violating the delay and capacity constraints. The proposed mapping algorithm can be enhanced so that it can intelligently create SFCs.

Fang et al. [130] formulated the Reliability-Aware VNF Placement (RAVP) problem as an integer linear programming. Then they proposed two heuristic algorithms for reliable protection: All-Node Protection Mechanism (ANPM) and Single Node Protection Mechanism (SNPM). ANPM provides protection to entire SFC, while SNPM provides protection to only single VNF. All existing work considered reliability of VNFs only, but the link reliability of SFC is missing. This scheme offers multiple path protection rather than a single path to improve SFC reliability.

5.3.1. Discussion

In AVailability Aware SFC provisioning (AVA-SFC), the proposed taxonomy considered fault recovery criteria of node failure, link failure, and whole SFP failure, as well as a fault detection strategy of single instance and multiple instances, to categorize resilient-aware solutions in literature. In the final set of articles, 21 articles deal with AVA-SFC. In this category of methods, the recovery of node and link failure is done by assigning the workload of failed node to another SF and choosing an alternative path for it. We observed that most of the researchers worked on either node or link failure, whereas in a real scenario, whole SFP has to recover from the failure. Hence, fault recovery of whole SFP is identified as an open research issue that needs to be explored. Moreover, the proposed solutions can be enhanced to handle a more complex scenario with the simultaneous failure of multiple VNFs and links. The existing approaches should also offer multiple path protection rather than a single path to improve SFC reliability.

6. RA4: Service function chaining challenges

A number of advantages are offered by dynamic service chaining, but it also comes with its own research challenges. The following are some of the research challenges faced during SFC deployment.

1. Dedicated Topology: The network topology is a core component to deploy any network functionality. Therefore, the dependency on network topology to deploy any network function restricts the redundancy, scalability, and resource utilization.

2. Complex Network Configuration: The complexity of network configuration depends upon the dedicated network topology for deployment of SFC. Any type of change (addition or deletion of network service) in existing chains requires a change in the ordering of chain, network topology and accordingly, there is a need to change the configuration of the chain. Due to this complexity, network operators do not want to change network topology once they have installed, configured and deployed the network.

3. Dynamic ordering of service functions: The number of services in the SFC is independent of each other, but current SFCs are rigid in nature because they are built based on manual configuration. In the static type of model, every packet or flow, will have to pass through the chain, even though some requests need only a subset of these network services [28]. The solution of this problem is solved by dynamic service chaining. In dynamic service chaining, SDN and NFV replace traditional middle-boxes with VMs and allow dynamic service chaining [33]. In dynamic service chaining, the traffic needs to be steered only through desired network functions according to specific flow requirements [132]. SDN controller can create chains dynamically and forward traffic intelligently to a particular network function based on the label such as VLAN, Source MAC address, or Network Service Header (NSH). This type of chain is called a software control service chain [94].

4. Security: The protection of SFC architecture from attacks is most prominent research challenge that need to be addressed for better performance [133]. The primary purpose of security is to ensure the SFC architecture is stable and robust [3]. The VNFs can be vulnerable to security attacks and whole service function chain can fail. Although in literature a limited number of solutions are provided, but still it is an open issue for the researchers. The paper [134] provided a survey on security challenges in NFV and discussed security solutions.

5. Resiliency: In order to improve the availability of network services, re-composition, re-mapping, and re-scheduling of the failed SFC should be automated. This mechanism should not impact the other service chains to maintain service continuity. In ref [114], authors presented the model for the high availability of SFCs by creating different backup strategies. The authors of [56] presented resilient orchestration feature of SFC that is able to create a new chain at any time as well as reroute the traffic to new SFP, if any fault appears. Karra, and Sivalingam [121] designed two algorithms to provide resilient services with minimum resource redundancy.

5. VNF placement: The optimal mapping of SFC to the substrate network is an open research topic that is not discussed enough in literature. The problem of optimally placing the network functions is called VNF placement or SFC resource allocation problem. The placement of new instantiated SFs or migrated SF instances, should be clearly investigated. The authors of the paper [106] modeled multi-stage graph and greedy algorithm for VNF placement. The proposed algorithms provide a time efficient solution by defining accessible scope for limiting the search space. Wang et al. [105] developed heuristic algorithm named Jcap to solve SFC-CPA (Composition, Placement, Assignment) problem.

7. RA5: Research gaps

The following are some of the research gaps that were found after comprehensive study of existing SFC provisioning strategies. Further exploration and investigation of these gaps will aid in the development of efficient and effective SFC solutions.

- Majority of researchers such as [63–66,69,71,74,76,77,91, 121] validated their SFC provisioning strategies using a mininet emulator. So, actual validation of the SFC strategies by creating VMs on real hardware is an open research issue.
- The LBA-SFC approaches such as [60,65,67,71,73,76,83] considered only VNF load balancing and link load balancing is ignored by the researchers. On the other hand, [48,82] validated only link load balancing. A combination of both VNF and link load balancing is an open research area which may lead to reducing latency and improving throughput.
- The PLA-SFC provisioning solutions such as [40,58,85,86,89, 90,93–100,104,105,108–110] considered only VNFs for their SFC. SFC provisioning strategy for PNFs is discussed by [91]. The other researchers such as [87,92,107] validated their SFC strategy using combination of VNFs and PNFs. But none of the works has discussed the combination of CNFs and VNFs to reduce the service response time.
- The AVA-SFC provisioning solutions such as [56,112,113, 115,116,118,120,121,124–126,128,129] recovered only from the node failure. The recovery of link failure has been discussed by [119]. To improve reliability, the fault recovery of whole service function path needs to be addressed.
- The researchers [48,60,63,64,73,76–78,83,86,87,89–91,93–96,98,99,99] have validated their proposed approaches with only static SFCs. But considering dynamic scenario in real world is also an open research area.

8. Conclusion

In the era of 5th generation mobile network, we can connect almost everything such as machines, objects, and devices together. 5G technology is being adopted as a global standard and promises high bandwidth, low latency, improved reliability, increased availability, and uniform user experience. NFV and SDN are complementary technologies that help overcome architectural challenges in deployment of 5G by providing capabilities such as network slicing and optimal flow management. An important factor in ensuring QoS delivery to users in these networks is the deployment of dynamic SFCs with the synergy of multiple factors viz. placement of VNFs, load balancing, and availability.

In this paper, a comprehensive and systematic survey of the SFC provisioning approaches has been presented. The most relevant research articles have been selected using Systematic Literature Review (SLR) protocol. A total of 70 articles published between 2015 to 2020 have been selected from IEEE Xplore, Science Direct, ACM and Google Scholar digital libraries. It has been found that around 30% researchers have used Load Balancing Aware SFC (LBA-SFC) techniques, about 40% have used Placement Aware SFC (PLA-SFC) approaches, and approximately 30% have used Availability Aware SFC (AVA-SFC) approaches. After critically reviewing these research articles, it is summarized that to provide the high quality, reliable, scalable, pragmatic and cost-effective SFCs, there is a need and demand of innovative solutions in placement of VNFs, load balancing, and providing availability. Further, an optimal amalgamate of these novel solutions with minimum overheads shall serve the requirements of next generation networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, Vyas Sekar, Making middleboxes someone else's problem: Network processing as a cloud service, ACM SIGCOMM Comput. Commun. Rev. 42 (4) (2012) 13–24.

[2] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, Raouf Boutaba, Network function virtualization: State-of-the-art and research challenges, IEEE Commun. Surv. Tutor. 18 (1) (2015) 236–262.

[3] Shin-ichi Kuribayashi, Allocation of virtual firewall functions in NFV-based networks with minimum network cost, Int. J. Comput. Netw. Commun. 11 (2) (2019) 1–15.

[4] Simon Grinberg, Shlomo Weiss, Architectural virtualization extensions: A systems perspective, Comp. Sci. Rev. 6 (5–6) (2012) 209–224.

[5] A.U. Rehman, Rui L. Aguiar, João Paulo Barraca, Network functions virtualization: The long road to commercial deployments, IEEE Access 7 (2019) 60439–60464.

[6] Jagdeep Singh, Sunny Behal, Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions, Comp. Sci. Rev. 37 (2020) 100279.

[7] Yong Li, Min Chen, Software-defined network function virtualization: A survey, IEEE Access 3 (2015) 2542–2553.

[8] Pankaj Berde, Matteo Gerola, Jonathan Hart, Yuta Higuchi, Masayoshi Kobayashi, Toshio Koide, Bob Lantz, Brian O'Connor, Pavlin Radoslavov, William Snow, et al., ONOS: Towards an open, distributed SDN OS, in: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, 2014, pp. 1–6.

[9] Deval Bhamare, Raj Jain, Mohammed Samaka, Aiman Erbad, A survey on service function chaining, J. Netw. Comput. Appl. 75 (2016) 138–155.

[10] Hajar Hantouti, Nabil Benamar, Tarik Taleb, Abdelquoddous Laghrissi, Traffic steering for service function chaining, IEEE Commun. Surv. Tutor. 21 (1) (2018) 487–507.

[11] Irena Trajkovska, Michail-Alexandros Kourtis, Christos Sakkas, Denis Baudinot, João Silva, Piyush Harsh, George Xylouris, Thomas Michael Bohnert, Harilaos Koumaras, SDN-based service function chaining mechanism and service prototype implementation in NFV scenario, Comput. Stand. Interfaces 54 (2017) 247–265.

[12] Xu Li, Jaya Rao, Hang Zhang, Aaron Callard, Network slicing with elastic SFC, in: 2017 IEEE 86th Vehicular Technology Conference, VTC-Fall, IEEE, 2017, pp. 1–5.

[13] Alcardo Alex Barakabitze, Arslan Ahmad, Rashid Mijumbi, Andrew Hines, 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges, Comput. Netw. 167 (2020) 106984.

[14] Michel S. Bonfim, Kelvin L. Dias, Stenio F.L. Fernandes, Integrated NFV/SDN architectures: A systematic literature review, ACM Comput. Surv. 51 (6) (2019) 1–39.

[15] Ahmed M Medhat, Tarik Taleb, Asma Elmangoush, Giuseppe A Carella, Stefan Covaci, Thomas Magedanz, Service function chaining in next generation networks: State of the art and research challenges, IEEE Commun. Mag. 55 (2) (2016) 216–223.

[16] Ghasem Mirjalily, Luo Zhiquan, Optimal network function virtualization and service function chaining: A survey, Chin. J. Electron. 27 (4) (2018) 704–717.

[17] Sedef Demirci, Seref Sagiroglu, Optimal placement of virtual network functions in software defined networks: A survey, J. Netw. Comput. Appl. 147 (2019) 102424.

[18] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, Seungjoon Lee, Network function virtualization: Challenges and opportunities for innovations, IEEE Commun. Mag. 53 (2) (2015) 90–97.

[19] Pooyan Jamshidi, Aakash Ahmad, Claus Pahl, Cloud migration research: A systematic review, IEEE Trans. Cloud Comput. 1 (2) (2013) 142–157.

[20] Muhammad Ali Babar, He Zhang, Systematic literature reviews in software engineering: Preliminary results from interviews with researchers, in: 2009 3rd International Symposium on Empirical Software Engineering and Measurement, IEEE, 2009, pp. 346–355.

[21] Karanpreet Singh, Paramvir Singh, Krishan Kumar, A systematic review of IP traceback schemes for denial of service attacks, Comput. Secur. 56 (2016) 111–139.

[22] Xin Li, Mohammed Samaka, H Anthony Chan, Deval Bhamare, Lav Gupta, Chengcheng Guo, Raj Jain, Network slicing for 5G: Challenges and opportunities, IEEE Internet Comput. 21 (5) (2017) 20–27.

[23] Fifth generation mobile network 5G, 2020, https://www.qualcomm.com/invention/5g/what-is-5g. (Accessed 17 March 2020).

[24] IEEEXplore Digital Library, 2019, https://ieeexplore.ieee.org/Xplore/home.jsp. (Accessed 26 June 2019).

[25] Springer digital library, 2019, https://link.springer.com/. (Accessed 20 June 2019).

[26] Sciencedirecct digital library, 2019, https://www.sciencedirect.com/. (Accessed 23 June 2019).

[27] ACM Digital library, 2019, https://dl.acm.org/. (Accessed 24 June 2019).

[28] Can Ouyang, Yunkai Wei, Supeng Leng, Yijin Chen, Service chain performance optimization based on middlebox deployment, in: 2017 IEEE 17th International Conference on Communication Technology, ICCT, IEEE, 2017, pp. 1947–1952.

[29] Bilal Anwer, Theophilus Benson, Nick Feamster, Dave Levin, Programming slick network functions, in: Proceedings of the 1st Acm Sigcomm Symposium on Software Defined Networking Research, 2015, pp. 1–13.

[30] Taixin Li, Huachun Zhou, Hongbin Luo, A new method for providing network services: Service function chain, Opt. Switch. Netw. 26 (2017) 60–68.

[31] Uwe G. Wilhelm, Sebastian M. Staamann, Levente Buttyán, A pessimistic approach to trust in mobile agent platforms, IEEE Internet Comput. 4 (5) (2000) 40–48.

[32] Zafar Ayyub Qazi, Cheng-Chun Tu, Luis Chiang, Rui Miao, Vyas Sekar, Minlan Yu, SIMPLE-fying middlebox policy enforcement using SDN, in: Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, 2013, pp. 27–38.

[33] Ola Salman, Imad H. Elhajj, Ayman Kayssi, Ali Chehab, SDN controllers: A comparative study, in: 2016 18th Mediterranean Electrotechnical Conference, MELECON, IEEE, 2016, pp. 1–6.

[34] Gursharan Singh, Sunny Behal, Monal Taneja, Advanced memory reusing mechanism for virtual machines in cloud computing, Procedia Comput. Sci. 57 (5) (2015) 91–103.

[35] Zili Ning, Ning Wang, Rahim Tafazolli, Deep reinforcement learning for NFV-based service function chaining in multi-service networks, in: 2020 IEEE 21st International Conference on High Performance Switching and Routing, HPSR, IEEE, 2020, pp. 1–6.

[36] H. Kitada, H. Kojima, N. Takaya, M. Aihara, Service function chaining technology for future networks, NTT Tech. Rev. 12 (8) (2014) 1–5.

[37] Cristina K. Dominicini, Gilmar L. Vassoler, Rodolfo Valentim, Rodolfo S. Villaca, Moisés R.N. Ribeiro, Magnos Martinello, Eduardo Zambon, KeySFC: Traffic steering using strict source routing for dynamic and efficient network orchestration, Comput. Netw. 167 (2020) 106975.

[38] Wenke Yan, Konglin Zhu, Lin Zhang, Sixi Su, Efficient dynamic service function chain combination of network function virtualization, in: 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops, ICDCSW, IEEE, 2017, pp. 163–168.

[39] Salvatore Renato Ziri, Ahmad Tajuddin Samsudin, Christophe Fontaine, Service chaining implementation in network function virtualization with software defined networking, in: Proceedings of the 5th International Conference on Communications and Broadband Networking, 2017, pp. 70–75.

[40] Abhishek Gupta, M. Farhan Habib, Uttam Mandal, Pulak Chowdhury, Massimo Tornatore, Biswanath Mukherjee, On service-chaining strategies using virtual network functions in operator networks, Comput. Netw. 133 (2018) 1–16.
[41] Ming Xia, Meral Shirazipour, Ying Zhang, Howard Green, Attila Takacs, SOLuTIoN: SDN-based OpticaL traffic steering for NFV, in: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, 2014, pp. 227–228.
[42] Emmanouil Fountoulakis, Qi Liao, Nikolaos Pappas, An end-to-end performance analysis for service chaining in a virtualized network, IEEE Open J. Commun. Soc. 1 (2020) 148–163.
[43] Internet engineering task force (IETF), 2020, https://www.ietf.org. (Accessed 21 February 2020).
[44] Guozhen Cheng, Hongchang Chen, Hongchao Hu, Zhiming Wang, Julong Lan, Enabling network function combination via service chain instantiation, Comput. Netw. 92 (2015) 396–407.
[45] Service function chaining (SFC) architecture, 2020, https://www.hjp.at/(de)/doc/rfc/rfc7665.html. (Accessed 20 January 2020).
[46] Seungik Lee, Sangheon Pack, Myung-Ki Shin, Service function path adaptation in SFC, in: 2017 19th International Conference on Advanced Communication Technology, ICACT, IEEE, 2017, pp. 510–513.
[47] Paul Quinn, Jim Guichard, Service function chaining: Creating a service plane via network service headers, Computer 47 (11) (2014) 38–44.
[48] Li-Der Chou, Chia-Wei Tseng, Hsin-Yao Chou, Yao-Tsung Yang, A SFC network management system in SDN, in: International Conference on Mobile and Wireless Technology, Springer, 2017, pp. 360–369.
[49] Service chaining in carrier networks, 2020, https://www.qosmos.com/wp-content/uploads/Service-Chaining-in-Carrier-Networks_WP_Heavy-Reading_Qosmos_Feb2015.pdf. (Accessed 22 January 2020).
[50] Lalit Chettri, Rabindranath Bera, A comprehensive survey on internet of things (IoT) toward 5G wireless systems, IEEE Internet Things J. 7 (1) (2019) 16–32.
[51] Gustavo Miotto, Marcelo Caggiani Luizelli, Weverton Luis da Costa Cordeiro, Luciano Paschoal Gaspary, Adaptive placement & chaining of virtual network functions with NFV-PEAR, J. Internet Serv. Appl. 10 (1) (2019) 3.
[52] Hye-Jin Ku, J.H. Jung, Gu-In Kwon, A study on reinforcement learning based SFC path selection in SDN/NFV, Int. J. Appl. Eng. Res. 12 (12) (2017) 3439–3443.
[53] Yong Li, Feng Zheng, Min Chen, Depeng Jin, A unified control and optimization framework for dynamical service chaining in software-defined NFV system, IEEE Wirel. Commun. 22 (6) (2015) 15–23.
[54] Nikolaos Petroulakis, Konstantinos Fysarakis, Andreas Miaoudakis, Konstantinos Ramantas, Panos Chatziadam, Christos Verikoukis, Service chaining using software-defined networks, in: Wiley 5G Ref: The Essential 5G Reference Online, Wiley Online Library, 2019, pp. 1–23.
[55] Hajar Hantouti, Nabil Benamar, A novel SDN-based architecture and traffic steering method for service function chaining, in: 2018 International Conference on Selected Topics in Mobile and Wireless Networking, MoWNeT, IEEE, 2018, pp. 1–8.
[56] Ahmed M. Medhat, Giuseppe A. Carella, Michael Pauls, Marcello Monachesi, Marius Corici, Thomas Magedanz, Resilient orchestration of service functions chains in a NFV environment, in: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN, IEEE, 2016, pp. 7–12.
[57] Sang-Min Shin, Gu-In Kwon, Real-time monitoring technique using SFC classifier in nfv environment, Int. J. Appl. Eng. Res. 12 (19) (2017) 8676–8680.
[58] Selma Khebbache, Makhlouf Hadji, Djamal Zeghlache, Virtualized network functions chaining and routing algorithms, Comput. Netw. 114 (2017) 95–110.
[59] Wen-Ping Lai, Kuan-Chun Chiu, Automatic deployment and dynamic scaling of NFV service chaining on Bare Metal (SCBM), in: 2018 IEEE Vehicular Networking Conference, VNC, IEEE, 2018, pp. 1–2.
[60] Alexandre Heideker, Ivan Zyrianoff, Carlos A. Kamienski, Profiling service function chaining behavior for NFV orchestration, in: 2018 IEEE Symposium on Computers and Communications, ISCC, IEEE, 2018, pp. 01020–01025.
[61] Hao Wang, Jens Schmitt, Load balancing-towards balanced delay guarantees in NFV/SDN, in: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), IEEE, 2016, pp. 240–245.
[62] Sikandar Ejaz, Zeshan Iqbal, Peer Azmat Shah, Bilal Haider Bukhari, Armughan Ali, Farhan Aadil, Traffic load balancing using software defined networking (SDN) controller as virtualized network function, IEEE Access 7 (2019) 46646–46658.
[63] Giwon Lee, Myeongsu Kim, Sukjin Choo, Sangheon Pack, Younghwa Kim, Optimal flow distribution in service function chaining, in: The 10th International Conference on Future Internet, 2015, pp. 17–20.
[64] Ahmed M. Medhat, Giuseppe Carella, Christian Lück, Marius-Iulian Corici, Thomas Magedanz, Near optimal service function path instantiation in a multi-datacenter environment, in: 2015 11th International Conference on Network and Service Management, CNSM, IEEE, 2015, pp. 336–341.
[65] Po-Ching Lin, Ying-Dar Lin, Cheng-Ying Wu, Yuan-Cheng Lai, Yi-Chih Kao, Balanced service chaining in software-defined networks with network function virtualization, Computer 49 (11) (2016) 68–76.
[66] Minh-Tuan Thai, Ying-Dar Lin, Yuan-Cheng Lai, A joint network and server load balancing algorithm for chaining virtualized network functions, in: 2016 IEEE International Conference on Communications, ICC, IEEE, 2016, pp. 1–6.
[67] Nabeel Akhtar, Ibrahim Matta, Yuefeng Wang, Managing NFV using SDN and control theory, in: NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, IEEE, 2016, pp. 1113–1118.
[68] Boston university recursive internetwork architecture (RINA) lab, 2020, http://csr.bu.edu/rina/. (Accessed 12 March 2020).
[69] Minh-Tuan Thai, Ying-Dar Lin, Po-Ching Lin, Yuan-Cheng Lai, Hash-based load balanced traffic steering on softswitches for chaining virtualized network functions, in: 2017 IEEE International Conference on Communications, ICC, IEEE, 2017, pp. 1–6.
[70] Mosab Hamdan, Bushra Mohammed, Usman Humayun, Ahmed Abdelaziz, Suleman Khan, M. Akhtar Ali, Muhammad Imran, Muhammad Nadzir Marsono, Flow-aware elephant flow detection for software-defined networks, IEEE Access 8 (2020) 72585–72597.
[71] Dongjin Hong, Jinyong Kim, Daeyoung Hyun, Jaehoon Paul Jeong, A monitoring-based load balancing scheme for network security functions, in: 2017 International Conference on Information and Communication Technology Convergence, ICTC, IEEE, 2017, pp. 668–672.
[72] Sang Il Kim, Hwa Sung Kim, A research on dynamic service function chaining based on reinforcement learning using resource usage, in: 2017 Ninth International Conference on Ubiquitous and Future Networks, ICUFN, IEEE, 2017, pp. 582–586.
[73] Ahmed M. Medhat, Giuseppe A. Carella, Michael Pauls, Thomas Magedanz, Orchestrating scalable service function chains in a NFV environment, in: 2017 IEEE Conference on Network Softwarization, NetSoft, IEEE, 2017, pp. 1–5.
[74] Minh-Tuan Thai, Ying-Dar Lin, Po-Ching Lin, Yuan-Cheng Lai, Towards load-balanced service chaining by hash-based traffic steering on softswitches, J. Netw. Comput. Appl. 109 (2018) 1–10.
[75] Gang Sun, Gungyang Zhu, Dan Liao, Hongfang Yu, Xiaojiang Du, Mohsen Guizani, Cost-efficient service function chain orchestration for low-latency applications in NFV networks, IEEE Syst. J. 13 (4) (2018) 3877–3888.
[76] Yi-Wei Ma, Jiann-Liang Chen, Jia-Yi Jhou, Adaptive service function selection for network function virtualization networking, Future Gener. Comput. Syst. 91 (2019) 108–123.
[77] Peilin Hong, Kaiping Xue, Defang Li, et al., Resource aware routing for service function chains in SDN and NFV-enabled network, IEEE Trans. Serv. Comput. (2018) 1–13.
[78] Gang Sun, Zhu Xu, Hongfang Yu, Xi Chen, Victor Chang, Athanasios V. Vasilakos, Low-latency and resource-efficient service function chaining orchestration in network function virtualization, IEEE Internet Things J. 7 (7) (2020) 5760–5772.
[79] Yunjie Gu, Yuxiang Hu, Yuehang Ding, Jie Lu, Jichao Xie, Elastic virtual network function orchestration policy based on workload prediction, IEEE Access 7 (2019) 96868–96878.
[80] Gang Sun, Zhenrong Chen, Hongfang Yu, Xiaojiang Du, Mohsen Guizani, Online parallelized service function chain orchestration in data center networks, IEEE Access 7 (2019) 100147–100161.
[81] Yunjie Gu, Yuehang Ding, Yuxiang Hu, Joint optimization of delay guarantees and resource allocation for service function chaining, IEICE Trans. Inf. Syst. 102 (12) (2019) 2611–2614.
[82] Seyeon Jeong, Heegon Kim, Jae-Hyoung Yoo, James Won-Ki Hong, Machine learning based link state aware service function chaining, in: 2019 20th Asia-Pacific Network Operations and Management Symposium, APNOMS, IEEE, 2019, pp. 1–4.
[83] Boutheina Dab, Ilhem Fajjari, Mathieu Rohon, Cyril Auboin, Arnaud Diquélou, An efficient traffic steering for cloud-native service function chaining, in: 2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops, ICIN, IEEE, 2020, pp. 71–78.
[84] Tatsuya Mori, Masato Uchida, Ryoichi Kawahara, Jianping Pan, Shigeki Goto, Identifying elephant flows through periodically sampled packets, in: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, 2004, pp. 115–120.

[85] Chaima Ghribi, Marouen Mechtri, Djamal Zeghlache, A dynamic programming algorithm for joint VNF placement and chaining, in: Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking, 2016, pp. 19–24.

[86] Gang Xiong, Yuxiang Hu, Julong Lan, Guozhen Cheng, A mechanism for configurable network service chaining and its implementation, KSII Trans. Internet Inf. Syst. 10 (8) (2016) 3701–3727.

[87] Marouen Mechtri, Chaima Ghribi, Djamal Zeghlache, A scalable algorithm for the placement of service function chains, IEEE Trans. Netw. Serv. Manag. 13 (3) (2016) 533–546.

[88] Shinji Umeyama, An eigendecomposition approach to weighted graph matching problems, IEEE Trans. Pattern Anal. Mach. Intell. 10 (5) (1988) 695–703.

[89] Luhan Wang, Zhaoming Lu, Xiangming Wen, Raymond Knopp, Rohit Gupta, Joint optimization of service function chaining and resource allocation in network function virtualization, IEEE Access 4 (2016) 8084–8094.

[90] Cheng-Liang Hsieh, Ning Weng, Virtual network functions instantiation on SDN switches for policy-aware traffic steering, in: Proceedings of the 2016 Symposium on Architectures for Networking and Communications Systems, 2016, pp. 119–120.

[91] Siri Kim, Yunjung Han, Sungyong Park, An energy-aware service function chaining and reconfiguration algorithm in NFV, in: 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems, FAS* W, IEEE, 2016, pp. 54–59.

[92] Oussama Soualah, Marouen Mechtri, Chaima Ghribi, Djamal Zeghlache, Energy efficient algorithm for VNF placement and chaining, in: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGRID, IEEE, 2017, pp. 579–588.

[93] Bo Yi, Xingwei Wang, Min Huang, Design and evaluation of schemes for provisioning service function chain with function scalability, J. Netw. Comput. Appl. 93 (2017) 197–214.

[94] Gianluca Davoli, Walter Cerroni, Chiara Contoli, Francesco Foresta, Franco Callegati, Implementation of service function chaining control plane through openflow, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN, IEEE, 2017, pp. 1–4.

[95] Yang Zhang, Bilal Anwer, Vijay Gopalakrishnan, Bo Han, Joshua Reich, Aman Shaikh, Zhi-Li Zhang, Parabox: Exploiting parallelism for virtual network functions in service chaining, in: Proceedings of the Symposium on SDN Research, 2017, pp. 143–149.

[96] Salvatore Renato Ziri, Ahmad Tajuddin Samsudin, Christophe Fontaine, Service chaining implementation in network function virtualization with software defined networking, in: Proceedings of the 5th International Conference on Communications and Broadband Networking, 2017, pp. 70–75.

[97] Nicolas Huin, Andrea Tomassilli, Frédéric Giroire, Brigitte Jaumard, Energy-efficient service function chain provisioning, J. Opt. Commun. Netw. 10 (3) (2018) 114–124.

[98] Yicen Liu, Yu Lu, Xingkai Chen, Xi Li, Wenxin Qiao, Liyun Chen, A dynamic placement mechanism of service function chaining based on software-defined networking, TIIS 12 (10) (2018) 4640–4661.

[99] Nicolas Huin, Brigitte Jaumard, Frédéric Giroire, Optimal network service chain provisioning, IEEE/ACM Trans. Netw. 26 (3) (2018) 1320–1333.

[100] Maryam Jalalitabar, Evrim Guler, Danyang Zheng, Guangchun Luo, Ling Tian, Xiaojun Cao, Embedding dependence-aware service function chains, J. Opt. Commun. Netw. 10 (8) (2018) C64–C74.

[101] Akshay Gadre, Anix Anbiah, Krishna M. Sivalingam, Centralized approaches for virtual network function placement in SDN-enabled networks, EURASIP J. Wireless Commun. Networking 2018 (1) (2018) 197.

[102] Yulun Cheng, Longxiang Yang, Hongbo Zhu, Deployment of service function chain for NFV-enabled network with delay constraint, in: 2018 International Conference on Electronics Technology, ICET, IEEE, 2018, pp. 383–386.

[103] Hadi Razzaghi Kouchaksaraei, Holger Karl, Service function chaining across openstack and kubernetes domains, in: Proceedings of the 13th ACM International Conference on Distributed and Event-Based Systems, 2019, pp. 240–243.

[104] Haotong Cao, Hongbo Zhu, Longxiang Yang, Dynamic embedding and scheduling of service function chains for future SDN/NFV-enabled networks, IEEE Access 7 (2019) 39721–39730.

[105] Zenan Wang, Jiao Zhang, Tao Huang, Yunjie Liu, Service function chain composition, placement, and assignment in data centers, IEEE Trans. Netw. Serv. Manag. 16 (4) (2019) 1638–1650.

[106] Dandan Qi, Subin Shen, Guanghui Wang, Towards an efficient VNF placement in network function virtualization, Comput. Commun. 138 (2019) 81–89.

[107] Oussama Soualah, Marouen Mechtri, Chaima Ghribi, Djamal Zeghlache, Online and batch algorithms for VNFs placement and chaining, Comput. Netw. 158 (2019) 98–113.

[108] Gang Sun, Yayu Li, Hongfang Yu, Athanasios V. Vasilakos, Xiaojiang Du, Mohsen Guizani, Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks, Future Gener. Comput. Syst. 91 (2019) 347–360.

[109] Gengbiao Shen, Qing Li, Yong Jiang, Yu Wu, Jianhui Lv, A four-stage adaptive scheduling scheme for service function chain in NFV, Comput. Netw. 175 (2020) 107259.

[110] Guanglei Li, Bohao Feng, Huachun Zhou, Yuming Zhang, Keshav Sood, Shui Yu, Adaptive service function chaining mappings in 5G using deep q-learning, Comput. Commun. 152 (2020) 305–315.

[111] Imane El Mensoum, Omar Abdul Wahab, Nadjia Kara, Claes Edstrom, MuSC: A multi-stage service chains embedding approach, J. Netw. Comput. Appl. 159 (2020) 102593.

[112] Seung-Ik Lee, Myung-Ki Shin, A self-recovery scheme for service function chaining, in: 2015 International Conference on Information and Communication Technology Convergence, ICTC, IEEE, 2015, pp. 108–112.

[113] Hongseok Jeon, BhumCheol Lee, Network service chaining challenges for VNF outsourcing in network function virtualization, in: 2015 International Conference on Information and Communication Technology Convergence, ICTC, IEEE, 2015, pp. 819–821.

[114] Sandra Herker, Xueli An, Wolfgang Kiess, Sergio Beker, Andreas Kirstaedter, Data-center architecture impacts on virtualized network functions service chain embedding with high availability requirements, in: 2015 IEEE Globecom Workshops, GC Wkshps, IEEE, 2015, pp. 1–7.

[115] Lyunsik Yang, Doan Van Tung, Minsik Kim, Younghan Kim, Alarm-based monitoring for high availability in service function chain, in: 2016 International Conference on Cloud Computing Research and Innovations, ICCCRI, IEEE, 2016, pp. 86–91.

[116] Shashank Bijwe, Fumio Machida, Shinya Ishida, Seiichi Koizumi, End-to-end reliability assurance of service chain embedding for network function virtualization, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN, IEEE, 2017, pp. 1–4.

[117] Shilei Zhang, Ying Wang, Wenjing Li, Xuesong Qiu, Service failure diagnosis in service function chain, in: 2017 19th Asia-Pacific Network Operations and Management Symposium, APNOMS, IEEE, 2017, pp. 70–75.

[118] Dongeun Suh, Hosung Baek, Seokwon Jang, Sol Han, Sangheon Pack, Distributed service function failover mechanism in service function chaining, in: 2017 International Conference on Information Networking, ICOIN, IEEE, 2017, pp. 148–150.

[119] Oussama Soualah, Marouen Mechtri, Chaima Ghribi, Djamal Zeghlache, A link failure recovery algorithm for virtual network function chaining, in: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management, IM, IEEE, 2017, pp. 213–221.

[120] Jaewook Lee, Haneul Ko, Dongeun Suh, Seokwon Jang, Sangheon Pack, Overload and failure management in service function chaining, in: 2017 IEEE Conference on Network Softwarization, NetSoft, IEEE, 2017, pp. 1–5.

[121] Karthik Karra, Krishna M. Sivalingam, Providing resiliency for service function chaining in NFV systems using a SDN-based approach, in: 2018 Twenty Fourth National Conference on Communications, NCC, IEEE, 2018, pp. 1–6.

[122] Jian Sun, Gang Sun, Dan Liao, Yao Li, Muthu Ramachandran, Victor Chang, Reliable and efficient deployment for virtual network functions, in: International Conference on Smart Computing and Communication, Springer, 2017, pp. 375–384.

[123] O.I. Aiko, Motomu Nakajima, Yuji Soejima, Mitsuho Tahara, Reliable design method for service function chaining, in: 2019 20th Asia-Pacific Network Operations and Management Symposium, APNOMS, IEEE, 2019, pp. 1–4.

[124] Pegah Torkamandi, Siavash Khorsandi, Bahador Bakhshi, Availability aware SFC embedding in NFV: A clustering approach, in: 2019 27th Iranian Conference on Electrical Engineering, ICEE, IEEE, 2019, pp. 1922–1928.

[125] Jiao Zhang, Zenan Wang, Chunyi Peng, Linquan Zhang, Tao Huang, Yunjie Liu, RABA: Resource-aware backup allocation for a chain of virtual network functions, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 1918–1926.

[126] Guanjun Tang, Yonghua Chen, Shiguang Xu, Qian Chen, Wenchen He, RSCM: A reliability-aware service chain mapping, in: International Conference on Artificial Intelligence and Security, Springer, 2019, pp. 653–662.

[127] Long Qu, Chadi Assi, Maurice J. Khabbaz, Yinghua Ye, Reliability-aware service function chaining with function decomposition and multipath routing, IEEE Trans. Netw. Serv. Manag. 17 (2) (2019) 835–848.
[128] Ngoc-Thanh Dinh, Younghan Kim, An efficient reliability guaranteed deployment scheme for service function chains, IEEE Access 7 (2019) 46491–46505.
[129] Yicen Liu, Yu Lu, Wenxin Qiao, Xingkai Chen, Reliability-aware service chaining mapping in NFV-enabled networks, ETRI J. 41 (2) (2019) 207–223.
[130] Lang Fang, Xiaoning Zhang, Keshav Sood, Yunqing Wang, Shui Yu, Reliability-aware virtual network function placement in carrier networks, J. Netw. Comput. Appl. 154 (2020) 102536.
[131] Meng Wang, Bo Cheng, Shuai Zhao, Biyi Li, Wendi Feng, Junliang Chen, Availability-aware service chain composition and mapping in NFV-enabled networks, in: 2019 IEEE International Conference on Web Services, ICWS, IEEE, 2019, pp. 107–115.
[132] ChiSheng Sua, Fu-Min Changb, Shang-Juh Kaoa, An OpenFlow-based dynamic service chaining approach for hybrid network functions virtualization, in: Proceedings of the 4th IIAE International Conference on Industrial Application Engineering, 2016, pp. 83–88.
[133] Kalpana D. Joshi, Kotaro Kataoka, pSMART: A lightweight, privacy-aware service function chain orchestration in multi-domain NFV/SDN, Comput. Netw. 178 (2020) 107295.
[134] Wei Yang, Carol Fung, A survey on security in network functions virtualization, in: 2016 IEEE NetSoft Conference and Workshops, NetSoft, IEEE, 2016, pp. 15–19.

Karamjeet Kaur received the B.Tech. degree from Punjabi University Patiala and M.Tech. degree from Punjab Technical University, Jalandhar in Computer Science Engineering. She is currently pursuing her Ph.D. in Information Technology from Panjab University Chandigarh. Her research interest is on Software-Defined Networking, Virtualization, Network Function Virtualization and Cloud Computing. She has published 22 papers in International & National Conferences. She has got 4 years teaching experience. She is the author of the 1 book. She is UGC-NET qualified.

Dr. Veenu Mangat received her Masters of Engineering in Computer Science and Engineering from Punjab Engineering College (PEC) in 2004 and Ph.D. in Engineering and Technology (Computer Science) in 2016 from Panjab University, India. She is currently working as Associate Professor in Information Technology at UIET, Panjab University. She has a teaching experience of more than 15 years. Her areas of research include data mining, machine learning, privacy and security. She is co-Principal Investigator in research project on 'Monitoring of Active Fire Locations and Precision in Allied Agricultural Activities using Communication Technologies' funded by Ministry of Electronics & IT of Government of India worth Rs. 75.75 lakhs from 2020–2022. She has also worked on research project entitled 'Pedestrian Detection from Thermal Imaging' funded by Design Innovation Centre of Ministry of HRD and consultancy project in the area of machine learning. She has successfully guided 21 Masters of Engineering dissertations and is currently guiding 7 Ph.D. scholars.

Dr. Krishan Kumar is currently Professor in Department of Information Technology, University Institute of Engineering and Technology, Panjab University, Chandigarh. He has done B. Tech. Computer Science & Engineering from National Institute of Technology, Hamirpur in 1995. He completed his Master of Software Systems from Birla Institute of Technology & Sciences, Pilani in 2001. He finished his regular Ph.D. from Indian Institute of Technology, Roorkee in February, 2008. He has more than 22 years of teaching, research and administrative experience. His general research interests are in the areas of Network Security and Computer Networks. Specific research interests include Intrusion Detection, Protection from Internet Attacks, Web performance, Network architecture/protocols, and Network measurement/ modeling. He has published 2 national and 2 International Books in the field of Computer Science & Network security. He has published more than 150 papers in national/International peer reviewed/Indexed/impact factor Journals and IEEE, ACM and Springer proceedings. His publications are well cited by eminent researchers in the field.