Digital Communications and Networks 10 (2024) 1057–1067

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

Position-aware packet loss optimization on service function chain placement

Wenjie Liang [a,1], Chengxiang Li [a,1], Lin Cui [a,*], Fung Po Tso [b]

[a] Department of Computer Science, Jinan University, Guangzhou, 510632, China
[b] Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, UK

ARTICLE INFO

Keywords:
Network function virtualization
Resource scheduling
SFC deployment
Packet loss

ABSTRACT

The advent of Network Function Virtualization (NFV) and Service Function Chains (SFCs) unleashes the power of dynamic creation of network services using Virtual Network Functions (VNFs). This is of great interest to network operators since poor service quality and resource wastage can potentially hurt their revenue in the long term. However, the study shows with a set of test-bed experiments that packet loss at certain positions (i.e., different VNFs) in an SFC can cause various degrees of resource wastage and performance degradation because of repeated upstream processing and transmission of retransmitted packets.

To overcome this challenge, this study focuses on resource scheduling and deployment of SFCs while considering packet loss positions. This study developed a novel SFC packet dropping cost model and formulated an SFC scheduling problem that aims to minimize overall packet dropping cost as a Mixed-Integer Linear Programming (MILP) and proved that it is NP-hard. In this study, *Palos* is proposed as an efficient scheme in exploiting the functional characteristics of VNFs and their positions in SFCs for scheduling resources and deployment to optimize packet dropping cost. Extensive experiment results show that *Palos* can achieve up to 42.73% improvement on packet dropping cost and up to 33.03% reduction on average SFC latency when compared with two other state-of-the-art schemes.

1. Introduction

Network Function Virtualization (NFV) leverages virtualization technology by making it possible to change the way of deploying network functions from dedicated hardware to software model. This greatly reduces both operational expenditures (OPEX) and capital expenditures (CAPEX) of Internet Service Providers (ISPs) [1,2]. With NFV, software network functions, also known as Virtual Network Functions (VNFs), can be managed more dynamically and efficiently. Once deployed, a collection of VNFs can be chained together in a specific order to provide customized network services in the form of Service Function Chains (SFCs).

SFC scheduling is of vital importance to network resource management. Because service providers need a robust scheduling scheme for SFCs to achieve operation goals such as lower cost and higher Quality of Experience (QoE) while providing innovative network services [3]. However, existing works mostly focus on VNF placement and migration [4–6] and largely assume that there is no packet loss when scheduling SFCs [7,8].

Previous studies [8–10] have observed that different types of VNFs vary in the aspect of processing capabilities and resource requirements. For example, allocating the same amount of CPU resources to two VNFs of different types may result in different processing rates. There is also an upper bound of processing rate for each VNF beyond which point allocating more resources to the VNF cannot bring any improvement in processing capabilities. These intrinsic characteristics of VNFs affect the decisions of SFC scheduling.

Furthermore, the position of VNFs in an SFC affects the scheduling of the SFC. Fig. 1 shows an example in which packet loss at different positions of an SFC results in very different impacts on performance. Because most VNF packet processing rates are determined by computing resources, VNF packet loss is usually caused by having insufficient computing resources allocated to it. When traffic flows arrive at a VNF with a certain arrival rate, a lower service rate of the VNF may cause queuing buffer overflow, leading to a higher probability of packet loss. Once packets are dropped at one of the downstream VNFs, packets that have already been processed by upstream VNFs of the SFC will need to be retransmitted, causing waste of computing resources on these VNFs as well as in network bandwidth. High ratio of packet loss can significantly impair the performance of network services enabled by SFCs. Therefore,

* Corresponding author.
E-mail address: tcuilin@jnu.edu.cn (L. Cui).
[1] The first two authors have contributed equally to this paper.

https://doi.org/10.1016/j.dcan.2023.01.002
Received 23 October 2021; Received in revised form 12 November 2022; Accepted 11 January 2023
Available online 13 January 2023

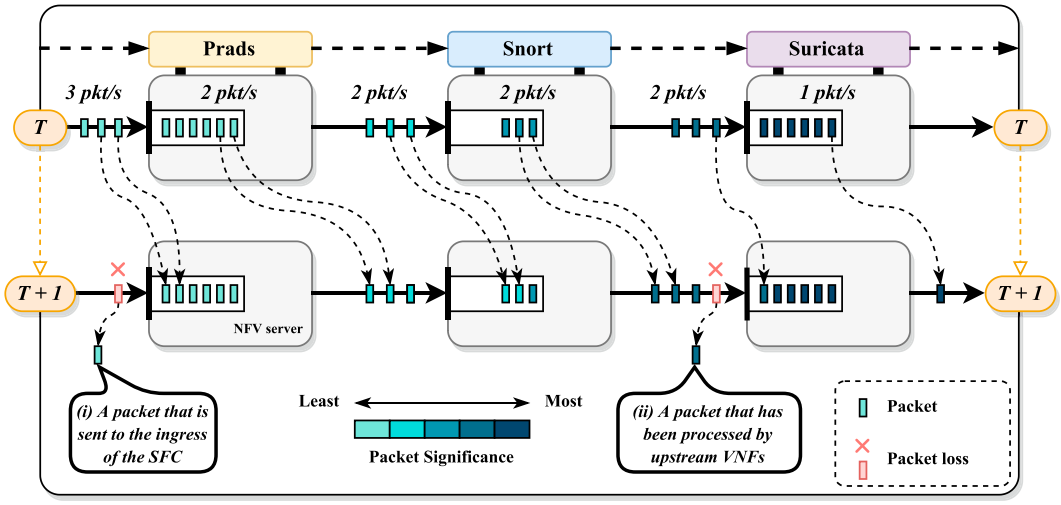


Fig. 1. A typical example of packets dropped at different positions of the SFC: *Prads* → *Snort* → *Suricata*. (i) A packet is dropped at the first VNF in an SFC; (ii) a packet that has been processed by the upstream VNFs (i.e., *Prads* and *Snort*) is dropped by the last VNF (i.e., *Suricata*). If both packets are retransmitted, the latter one needs to be processed all over again, resulting in extra computation and bandwidth consumption. (Higher significance level of a packet implies it has cost more bandwidth and computing resources.)

a reasonable resource scheduling scheme for SFCs is needed to alleviate resource wastage as a result of packet loss and improve the performance of SFCs.

In this study, both of the processing characteristics of VNFs and their positions in SFCs are considered to optimize the scheduling of SFCs. First, a set of test-bed experiments is conducted to investigate how resource allocation to VNFs at different SFC positions affects the performance including resource wastage of upstream processing and transmission. Based on these observations, a novel packet dropping cost is modeled and an efficient scheme *Palos* (**P**osition-**a**ware packet **l**oss **o**ptimization on **s**ervice function chains placement), is proposed.

The main contributions of this study are summarized as follows.

- This study investigates the impact of VNF positions in SFC and resource allocation on the performance of SFCs showing their high relevance through extensive test-bed experiments and analysis.
- The SFC scheduling problem, including resource allocation and placement of VNFs, is modeled by considering both positions of VNFs in SFC and characteristics of different types of VNFs. Particularly, the packet dropping cost as a major optimization objective has been formally defined and the problem is proven to be NP-hard.
- *Palos*is proposed as a scheme for scheduling resource and deployment with sensitivity to characteristics of VNFs and their positions in SFCs. Extensive evaluations demonstrate that *Palos* can achieve up to 42.73% improvement on reducing the packet dropping cost and reduce average SFC latency by up to 33.03% while ensuring bandwidth requirements of SFCs.

The remainder of this paper is organized as follows. In Section 2, motivations for the study is demonstrated through several test-bed experiments. Section 3 provides the formulation of resource scheduling and SFC deployment problems. In Section 4, *Palos*, an effective scheme to resource scheduling and deployment of SFCs is proposed in detail. In Section 5, extensive experiments are conducted to show the effectiveness of *Palos*. Section 6 presents an overview of related works, and Section 7 concludes the paper.

2. Motivating experiments

In this section, a series of test-bed experiments is conducted to investigate the resource requirements with respect to CPU allocation for different types of VNFs. To throttle the performance of an individual VNF at different positions of an SFC, the resource allocation was varied to different VNFs revealing the impact of packet loss on overall system performance.

Table 1
VNFs used in experiments.

VNF	Version	Description
Snort	2.9.16	Open source intrusion protection system
Prads	0.3.0	Passive real-time asset detection system
Suricata	4.1.8	Open source threat detection engine

2.1. Overview and experiment setup

Packet loss is one of the main factors that can degrade the performance of SFCs. As shown in Fig. 1, any packet dropped by VNFs at the later stage of the SFC can cause a greater level of wastage in both computing and network resources since lost/dropped packets will need to be reprocessed and retransmitted by VNFs in the early stage of the SFC. This will potentially create higher service latency which will not only impair the Quality of Service (QoS) but also risk violating Service-Level Agreements (SLAs). Hence, the packet loss positions have considerable impact on performance tuning for deployed SFCs. Packet loss in a VNF is usually caused by insufficient computing resources allocated to that VNF. Preliminary studies of this work discovered that resource allocation involves two key factors: *different properties of VNFs* and *positions of VNFs in SFC*.

To investigate the importance and impacts of these two factors on SFC performance, an experiment test-bed based on an NFV server with 8-core Intel(R) Core(TM) i7-4771 CPU @ 3.50 GHz and 24 GB memory was conducted. The OS of the server is Ubuntu 21.04. Three popular open source VNFs are implemented, as shown in Table 1. Each VNF is deployed in different *namespaces* to avoid interference, and *cgroups* are used to control the amount of computing resources assigned to each VNF. Furthermore, the *iperf3*[2] is used to send packets.

For VNFs in experiments, *Snort*[3] is configured in Network Intrusion

[2] https://iperf.fr/.
[3] https://www.snort.org/.

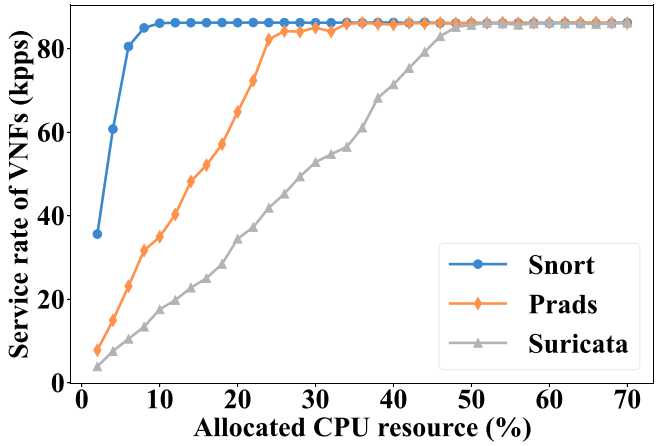


Fig. 2. Impact of allocated CPU resource on packet processing rates of different types of VNFs.

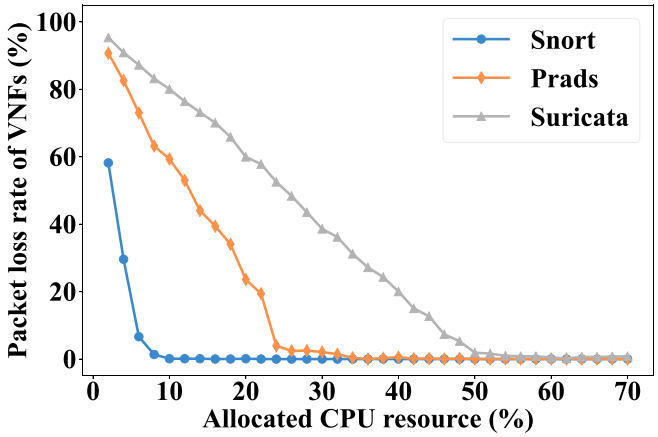


Fig. 3. Packet loss rates of different types of VNFs under allocated CPU resource.

Detection System (NIDS) mode that defines a set of rules and takes corresponding actions for matched packets. To avoid confusion from packet dropping due to predefined rules when analyzing experiment results, only matched packets are alerted and logged. *Prads*[4] is a real-time asset detection system that passively listens to network traffic and gathers information about hosts and services sending traffic. *Suricata*[5] runs in Network Security Monitor (NSM) mode.

2.2. Problem exploration & analysis

2.2.1. Performance of individual VNF

The first experiment investigates how varying CPU resources affect the performance of different types of VNFs. CPU resource is constrained from 2% to 70% with an increment of 2% for each VNF.

CPU vs. throughput. Fig. 2 shows the relationship between the packet processing rate of VNFs and allocated CPU resources. Because of their different internal packet processing mechanisms, different VNFs have varying processing rate even when they have the same allocated amount of CPU resources. For example, multi-threading allows quicker inspection on packets in the IDS mode for *Snort*. Moreover, Fig. 2 reveals that there are upper bounds in processing rates for each VNF, which means packet processing rates will not increase any further even if more CPU resources are allocated.

CPU vs. packet loss rate. Fig. 3 shows that packet loss rates of VNFs are influenced by allocated CPU resources. When these VNFs are allocated a small amount of resources (nearly 2%) severe packet dropping occurs since traffic cannot be timely processed. Moreover, packet loss rates decrease as more CPU resources are allocated.

As more CPU resources are allocated to VNFs, packet processing rates reach maximum value wherein packet loss rates approach zero. In other words, one VNF can reach its saturated state with a certain amount of allocated resources. Figs. 2 and 3 also show that, for a specific VNF, packet processing rate and packet loss rate are approximately related linearly to its allocated resource. As seen from both figures, the slopes of three lines vary from one to another. This is because of the unique packet processing mechanisms of different VNFs. For example, computation-intensive VNFs usually require more CPU resources.

Observation 1: Different types of VNFs with the same amount of allocated resource have various packet processing rates and loss rates.

Table 2
CPU resource allocation in four scenarios.

	Prads(%)	*Snort*(%)	*Suricata*(%)
Scenario 1	45	20	35
Scenario 2	20	20	60
Scenario 3	30	20	50
Scenario 4	30	15	55

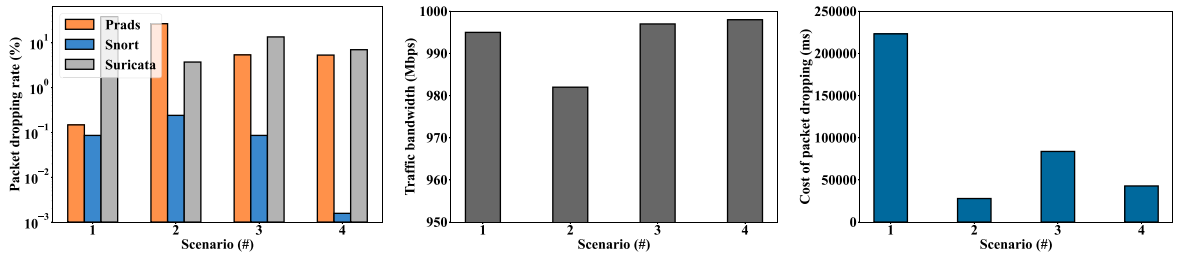
2.2.2. Performance of SFC

To evaluate the SFC performance, further experiments are conducted with an SFC comprising: *Prads* → *Snort* → *Suricata*. Assuming that SFC bandwidth requirement is no less than 990 Mbps, the total amount of resource of the server is 100%. To investigate impacts of VNFs' positions in SFC on resource allocation strategy, four scenarios with different combinations in the amount of CPU resources for the three VNFs are evaulated, as shown in Table 2. Three performance metrics are considered: packet loss rate, bandwidth and packet dropping cost. Packet dropping cost is defined as the processing time on all upstream VNFs before a packet is dropped, including both upstream VNFs' average service latency and propagation latency. The results of each scenario are averaged from 10 groups of experiments and each group of experiments runs for 600 *s*.

Fig. 4a shows packet dropping rates for each VNF in different scenarios. In Scenario 1, insufficient resources are allocated to the last VNF (i.e., *Suricata*) resulting in severe packet loss. In contrast, the first two VNFs have only 0.1% packet loss rates because they have been allocated sufficient resources. Packets dropped by *Suricata* can lead to more resource wastage because they are already processed by upstream VNFs and transmitted. Fig. 4c shows Scenario 1 has the highest packet dropping cost. In Scenario 2, when more resources are allocated to the last VNF, both packet dropping cost and packet loss rate decrease. However, the first VNF begins to drop packets due to insufficient resource allocation. In Scenario 3, 10% of CPU resources are taken away from the third VNF and allocated to the first VNF to improve the performance of Scenario 2. Scenario 4 demonstrates the best resource allocation strategy. From Fig. 3, the second VNF (i.e., *Snort*) is allocated with only 12% CPU resources to avoid packet loss, which implies that 20% CPU resources are too much for *Snort*. Therefore, 5% CPU resources can be safely taken away and allocated to the last VNF to give it an overall 55% CPU resource. As a result, overall packet loss rate is significantly reduced.

Observation 2: Insufficient allocation of resources to VNFs in the rear part of the SFC will result in higher packet loss and therefore, a higher packet dropping cost.

[4] https://github.com/gamelinux/prads.
[5] https://suricata-ids.org/.



(a) Packet dropping rates of VNFs. (b) Bandwidth of SFCs. (c) Packet dropping cost.

Fig. 4. Performance evaluation on an SFC chained by *Prads*, *Snort* and *Suricata* sequentially in different scenarios.

As shown in Fig. 4a and b, the packet loss rate of the last VNF in SFC is significantly reduced in Scenario 2 when compared to Scenario 1 but the bandwidth of the SFC is only about 985 Mbps which is the lowest among all scenarios. The reason is that the first VNF drops too many packets due to insufficient resources allocated to that VNF and thereby becomes the bottleneck of the SFC. Assuming the minimum bandwidth requirement is 990 Mbps, Scenario 2 will violate the SLAs. Scenarios 3 and 4 have higher throughput as *Prads* is allocated with 10% more resources. However, its packet dropping costs go up to 3 times compared to Scenario 2, as shown in Fig. 4c.

Observation 3: Simply allocating more resources to VNFs in the rear part of SFC does not guarantee higher performance as SFC throughput may be compromised.

As previously mentioned, different resource allocation to VNFs can cause different degrees of packet dropping. Specifically, when packet dropping happens in the downstream VNFs these packets have already been processed by the upstream VNFs in the SFC and transmitted, as explained in Fig. 1. Hence, different positions of packet dropping result in different levels of resource wastage. Fig. 4c demonstrates that Scenario 1 has the highest packet dropping cost as a large number of packets are dropped by the last VNF. Scenario 2 prevents severe packet loss in the last VNF and has the lowest packet dropping cost. However, too few resources are allocated to the first VNF, which throttles the throughput of the SFC. Moreover, acket dropping cost in Scenario 4 is obviously less than that in Scenario 3. To sum up, Scenario 4 provides the best resource allocation strategy and meets the SLAs requirement.

Observation 4: The performance of an SFC depends on the resource allocated to each VNF based on their positions in the SFC, and the characteristics of each VNF.

These experiments show that the orchestration of SFCs is complicated and challenging, especially when VNFs are deployed on different servers. This requires reasonable fine-grained resource scheduling for SFCs, especially when computing resources are limited. A trade-off between allocated resources and VNFs' positions in the SFC is inevitable whenever making SFC scheduling decisions.

Table 3
Key notations used in this paper.

Network	Description
$G = (V, E)$	A network consisting of a set of NFV servers V and a set of paths E between two NFV servers
$C(v)$	Available computing resource of a server $v \in V$
$\varphi(u, v)$, $\alpha(u, v)$	Bandwidth and propagation delay of path $(u, v) \in E$
VNF	**Description**
P	A set of types of VNFs
$k(f)$	Slope parameter of VNF f, which is equivalent to k_p
$\mu_{\max}(p)$, $r_{\max}(p)$	Maximum packet processing rate and corresponding resource requirement of VNF of type $p \in P$
$N(f)$	Buffer size of VNF f
$\lambda(f)$, $\mu(f)$, $r(f)$	Packet arrival rate, processing rate and the amount of CPU resources allocated to VNF f
$\rho(f)$	Utilization of VNF f
$r_{\min}(f)$	Minimum resource allocated to VNF f to satisfy the bandwidth requirement of SFC
$L(f)$, $\beta(f)$	Number of packets dropped and service latency of VNF f
$Prob_0(f)$, $Prob_N(f)$	Probabilities of that VNF queue is empty and full, respectively
SFC	**Description**
S	A set of SFCs, each of which contains an ordered collection of VNFs
b_i, w_i	Bandwidth requirement and expected path latency of SFC s_i
Other	**Description**
$X_{f,v}$	Binary variable that equals 1 if VNF f is located on server v
$Y^{i,j}_{u,v}$	Binary variable that equals 1 if virtual path from the j-th to $j+1$-th VNFs is deployed on physical path (u, v) in s_i

3. Problem model

In this section, the packet dropping cost is formally modeled and an SFC scheduling problem is formulated to minimize packet dropping cost across all SFCs. List of notations used in this paper is shown in Table 3.

3.1. Physical network

The physical network is defined as $G = (V, E)$, where V is a set of NFV servers and E is a set of physical paths between two NFV servers u and v. Previous studies have shown that CPU powered computing resources tend to be bottlenecks for VNFs deployed on a single server [11]. Thus, for simplicity, this study mainly focuses on CPU resources with an assumption that the memory resource of servers is sufficient. The available computing resource of a server $v \in V$ is denoted by $C(v)$. The bandwidth and propagation delay of a path $(u, v) \in E$ are denoted by $\varphi(u, v)$ and $\alpha(u, v)$, respectively.

3.2. Virtual Network Functions

There are different types of VNFs such as firewalls, intrusion detection systems (IDSs) and proxy servers. The set of types of VNFs to be deployed is defined as P. As observed in Section 2, different types of VNFs vary in the aspect of packet processing rate and resource requirement. For each VNF of type $p \in P$, its maximum packet processing rate and corresponding resource requirement are denoted by $\mu_{\max}(p)$ and $r_{\max}(p)$ respectively. Moreover, F represents a set of VNFs, where $f \in F$ is uniquely mapped to $p \in P$. Equation (1) shows the relationship between allocated resources and packet processing rate where VNF f is of type p. The slope parameter k_p varies with the types of VNFs. For each VNF, $N(f)$ denotes the buffer size of f, $\lambda(f)$ denotes packet arrival rate to f and $\mu(f)$ is packet processing rate.

$$\mu(f) = \begin{cases} k_p \cdot r(f), & r(f) < r_{max}(p) \\ \mu_{max}(p), & r(f) \geq r_{max}(p) \end{cases} \tag{1}$$

The value of $\mu(f)$ is determined by the VNF packet processing procedure. For example, VNFs that need to modify packet header or monitor

packet payload usually consume more computing resources. Furthermore, packet processing rates are also influenced by allocated resources, and VNFs with complex operations on packets have lower processing rates.

The position of VNF f in the network is shown in Equation (2): $X_{f,v}$ is a binary variable that represents whether VNF f is located on server v.

$$X_{f,v} = \begin{cases} 1, & \text{if } f \in F \text{ is located on server } v \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

3.3. Service function chains

An SFC is an ordered collection of multiple VNFs that performs ordered actions on network packets when traffic flows are steered through the SFC. The set of SFCs is defined as S. There are four properties for each SFC $s_i \in S$, i.e., $ingress_i$, $egress_i$, b_i and w_i. The ingress and egress of SFC s_i is marked by $ingress_i$ and $egress_i$ while b_i refers to bandwidth requirement and w_i is the expected path latency. Furthermore, $f_{i,j}(p)$ is the j-th VNF of type p in SFC s_i. Without loss of generality, shareable VNF is regarded as a replication of multiple VNF instances. Previous studies [12] have demonstrated this equivalence is feasible in actual deployment as total resources consumed by shareable VNFs equal the sum of consumed resources of multiple VNF instances.

To describe SFC deployment across different servers, a binary variable $Y_{u,v}^{i,j}$ is used to represent whether virtual path from the j-th to $j+1$-th VNFs is deployed on physical path (u, v) in s_i, as shown in Equation (3).

$$Y_{u,v}^{i,j} = \begin{cases} 1, & \text{if virtual path is on } (u,v) \in E \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

3.4. Packet dropping cost in SFC

Traffic flows arrive at the first VNF with a certain arrival rate, which may drop due to insufficient allocated resources to subsequent VNFs. Hence, the arrival rate to the j-th VNF is affected by previous $j-1$ VNFs in the upstream of the SFC. Some VNFs may change flow rate intentionally, e.g., rate limiter and traffic shaper wherein such behavior will complicate the problem (e.g. customized configurations of VNFs should be considered). Thus, they are out of the scope of this study, although there has been some research about such type of VNFs [13,14].

3.4.1. Resource allocation

To achieve high SFC performance when deploying a new VNF on a server, both characteristics of VNF types and packet arrival rates need to be considered when allocating limited computing resources. Experimental studies in Section 2 have shown that SFC performance is related to resources allocated to each VNF as well as their positions in the SFC. Moreover, its processing mechanism varies based on types of VNFs as does its resource requirement for the same arrival rate. All these factors should be considered for minimizing packet dropping cost while meeting the performance requirement of SFCs.

Denote $r(f)$ to be the amount of CPU resources in percent (%) allocated to VNF f. For each NFV server v, the sum of allocated resource to all VNFs in the server cannot exceed the capacity of the server, as shown in Equation (4):

$$\sum_{f \in F} r(f) \cdot X_{f,v} \le C(v), \forall v \in V \tag{4}$$

According to Section 2, the service rate of a VNF is directly influenced by allocated resources. Supposing the time of scheduling period is T, the total execution time of VNF f within a single scheduling period is $r(f) \cdot T$. The following equation represents the packet processing rate $\mu(f)$ for VNF f of type p. This equation can be simplified as: $\mu(f) = k_p \cdot r(f)$ as shown in Equation (1).

$$\mu(f) = \frac{\Theta(r(f)) \cdot T \cdot \mu_{max}(p)}{T} \cdot \frac{1}{\Theta(r_{max}(p))} \tag{5}$$

$$r_{min}(f) \le r(f) \le r_{max}(p) \tag{6}$$

$\Theta(r(f))$ is the actual amount of computing resources used for processing packets when the VNF f is allocated resource $r(f)$. If there are remaining resources after allocating $r(f)$ to VNF f these are evenly allocated to all VNFs on the server. Nevertheless, $\Theta(r(f))$ is strictly less than or equal to $r_{\max}(f)$. Furthermore, $\Theta(r(f)) \cdot T \cdot \mu_{\max}(p)$ is the total number of processed packets within a scheduling period T. $\frac{1}{\Theta(r_{max}(p))}$ is the maximum CPU utilization of VNF of type p. $\frac{1}{\Theta(r_{max}(p))} = 1$ implies that the peak value of packet processing rate can be achieved when CPU is 100% allocated. $\frac{\Theta(r(f))}{\Theta(r_{max}(p))}$ from Equation (5) is the ratio of resource allocated when the VNF achieves its maximum CPU utilization.

From Equation (5), we refer to $k_p \cdot r(f)$ as the shorthand of the equation, which represents actual packet processing rate of VNF f when CPU resource is $r(f)$. The value of k_p is determined by VNF type p. As mentioned in Section 2, different types of VNFs have different packet processing rates despite allocating the same amount of CPU resources, i.e., different slopes implying how fast packet processing rate is changing with respect to allocated resource. Constraint (6) shows that for each VNF f, allocated resource should not exceed $r_{\max}(p)$, i.e., its maximum processing rate cannot exceed the maximum value $\mu_{\max}(p)$. Moreover, $r_{\min}(f)$ represents the minimum amount of resource allocated to VNF f when satisfying the minimum requirement of bandwidth for the SFC. Since the bandwidth of downstream VNFs is influenced by the processing rate of upstream VNFs, the packet processing rate should not be lower than the bandwidth requirement of an SFC.

Lemma 1. *Packet processing rate $\mu(f)$ is positively correlated to the amount of allocated resource $r(f)$.*

Proof: Assuming that VNF f is allocated resource $r(f)$, $r_{\triangle}$ is the amount of CPU resource when there are no packets being processed, $r_{\diamond}$ is the amount of resource used for packet processing with respect to $r_{\max}(p)$. Both $r_{\triangle}$ and $r_{\diamond}$ are constants, which are determined by the type of VNF. In this case, formula of $\mu(f)$ can be transformed into:

$$\begin{aligned} \mu(f) &= \frac{(r(f) - r_{\triangle}) \cdot T \cdot \mu_{max}(p)}{T} \cdot \frac{1}{r_{max}(p) - r_{\diamond}} \\ &= \frac{\mu_{max}(p)}{r_{max}(p) - r_{\diamond}} \cdot r(f) - \frac{r_{\triangle} \cdot \mu_{max}(p)}{r_{max}(p) - r_{\diamond}} \end{aligned}$$

When $r(f)$ equals 0, $r_{\triangle}$ also equals 0, and $\frac{\mu_{max}(p)}{r_{max}(p) - r_{\diamond}}$ is a constant. Under any circumstances, $r(f) \gg r_{\triangle}$, then $\mu(f)$ is greater than zero. As such, with different allocated resource amounts, processing rate of the VNF $\mu(f)$ can equivalently be represented as $k_p \cdot r(f)$.

3.4.2. Packet dropping resource overhead

When traffic flows are processed by SFCs, VNFs may suffer different degrees of packet loss. Previous studies have shown that traffic flows generated by network service requests can be simulated as Poisson streams [15,16]. Assuming that the time interval for any two consecutive packets arriving at a VNF follows exponential distribution and the arrival process of each packet is independent, packets are buffered in a queue when they arrive at a VNF and are then independently processed at a certain service rate [17] following the first-come, first-served (FCFS) rule. Thus, the VNFs can be modeled as $M/M/1/N/\infty$ queues [18]. For each VNF $f \in F$, $\rho(f)$ refers to the utilization of the VNF, which is the ratio of arrival rate to service rate, i.e., $\frac{\lambda(f)}{\mu(f)}$. In a steady state, the probabilities [18] of VNF queue that is empty and full are given by Equation (7) and Equation (8), as follows:

$$Prob_0(f) = \frac{1-\rho(f)}{1-\rho^{N(f)+1}(f)} \tag{7}$$

$$Prob_N(f) = \rho^{N(f)}(f) \cdot Prob_0(f) \tag{8}$$

$Prob_N$ is the probability of rejecting packets. According to the $M/M/1/N/\infty$ queuing model [18], invalid arrival rate of packets can be equivalent to $\lambda \cdot Prob_N$. Given a time period ΔT, the number of dropped packets $L(f)$ caused by VNF f can be expressed as

$$L(f) = \lambda(f) \cdot Prob_N(f) \cdot \Delta T \tag{9}$$

Besides, λ' represents the valid arrival rate of packets. The probability that packets are processed by the VNF is $\lambda \cdot (1 - Prob_N)$. Furthermore, overload of the VNF is $\frac{\lambda'}{\mu}$, which reflects how busy the VNF is. Based on *Little's law* [18], the average service latency of VNF f, including queuing delay and processing delay, can be expressed as

$$\begin{aligned}\beta(f) &= \frac{\frac{\rho(f)}{1-\rho(f)} - \frac{(N(f)+1)\cdot\rho^{N(f)+1}(f)}{1-\rho^{N(f)+1}(f)}}{\lambda'(f)} \\ &= \frac{1}{\mu(f)} + \frac{\frac{\rho(f)}{1-\rho(f)} - \frac{(N(f)+1)\cdot\rho^{N(f)+1}(f)}{1-\rho^{N(f)+1}(f)} - 1 + Prob_0(f)}{\lambda'(f)}\end{aligned} \tag{10}$$

Packets dropped by a VNF have already been processed and transmitted by upstream VNFs in the SFC leading to wastage of these resources. This study uses latency as a key indicator to measure resources wasted due to packet loss. Therefore, for each VNF $f_{i,j} \in F$, its cost due to dropped packets is defined as the sum of upstream VNFs' average service latency and propagation latency for the packet dropped by $f_{i,j}$:

$$Cost_{i,j} = \sum_{n\in[1,j-1]} \left[\beta(f_{i,n}) + \sum_{(u,v)\in E} Y_{u,v}^{i,n} \cdot \alpha(u,v) \right]$$

3.5. SFC scheduling problem

The SFC scheduling problem is defined in Equation (11a). Its goal is to minimize packet dropping cost by reasonably scheduling resources among VNFs when deploying SFCs.

$$\min \sum_{i\in[1,|S|]} \sum_{j\in[1,|s_i|]} L(f_{i,j}) \cdot Cost_{i,j} \tag{11a}$$

$$s.t. \sum_{i\in[1,|S|]} \sum_{j\in[1,|s_i|]} Y_{u,v}^{i,j} \cdot b_i \le \varphi(u,v), \forall (u,v) \in E \tag{11b}$$

$$k(f_{i,j})) \cdot r(f_{i,j}) \ge b_i, \forall i \in [1,|S|], j \in [1,|s_i|] \tag{11c}$$

$$\sum_{(u,v)\in E} Y_{u,v}^{i,j} \cdot \alpha(u,v) \le w_i, \forall i \in [1,|S|], j \in [1,|s_i|] \tag{11d}$$

Deployment of SFCs should satisfy SLAs, including bandwidth requirement and expected propagation delay. Constraint (11b) is the bandwidth constraint for all SFC paths and $k(f_{i,j}) \cdot r(f_{i,j})$ in Constraint (11c) is the service rate (or throughput) of VNF $f_{i,j}$, where $k(f_{i,j})$ is equivalent to k_p as mentioned above. This constraint guarantees that resources allocated to any VNFs should be sufficient such that the service rate of the VNF is no less than the bandwidth requirement of the SFC. When the arrival rate of traffic flows remains unchanged, the amount of computing resources allocated to a VNF directly influences packet processing rate and further throughput.

Furthermore, propagation delay needs to be considered to avoid violation of SLAs because of higher latency of SFCs. Constraint (11d) ensures that the requirement of path latency for SFCs is met. As aforementioned, the problem in the deployment of SFCs is modeled as MILP.

Theorem 1. *The SFC deployment is an NP-hard problem.*

Proof: To prove the NP-hardness of this problem, the Multiple Knapsack Problem (MKP) [19] is introduced, which has already been proven to be NP-complete. MKP can be reduced to the SFC deployment problem in polynomial time. MKP describes that there exists $|M|$ items where each item $m_i \in M$ has a capacity requirement $req(m_i)$. Moreover, there exists $|D|$ bags with limited capacity $cap(d_i)$ for each $d_i \in D$. The profit of placing item m_i in a bag is $prof(m_i)$. The objective is to maximize total profit by packing these items into bags.

Similarly, for SFC deployment, the set of servers V can be regarded as bags D with limited capacity. $C(v)$ is equivalent to $cap(d_i)$. VNF $f_{i,j} \in F$ is equivalent to an item $m_i \in M$ and allocated resource $r(f_{i,j})$ is equivalent to $req(m_i)$. In addition to the location of VNFs, the amount of resources allocated to VNFs are also decision variables. Nevertheless, a VNF allocated with different amounts of resources can be viewed as multiple items of different weights to be packed. In this study, packet dropping cost is related to allocated resources, i.e., where to place VNFs determines packet dropping cost. In a special case where all SFCs contain only one VNF, the negative value of the cost of deploying $f_{i,j} \in F$ in server $v \in V$ can be treated as $prof(m_i)$. To summarize, the SFC deployment problem can be converted to the general MKP, with an objective to minimize total cost. Hence, MKP can be reduced to the SFC deployment problem in polynomial time and the former is proven to be NP-hard. □

4. *Palos* design

In this section, *Palos*is proposed as a scheme to deploy and schedule SFCs by considering both characteristics of VNFs and their positions in SFCs.

4.1. Analysis

Equations (8) and (10) show the probabilities that buffer of a VNF is full (i.e., $Prob_N$) and service latency of packets in the VNF decreases as service rate μ increases. Since packets will be dropped when the queue is full, $Prob_N$ determines the probability of packets dropped by VNFs. Thus, optimizing both packets dropped in the queue (including their positions) and processing latency of VNFs is important to improve performance of SFCs.

The arrival rate λ of a VNF is affected by the packet sending rate of the flow and amount of resource for upstream VNFs in the SFC. Intuitively, resource allocated for downstream VNFs of the SFC needs to be sufficient while the bandwidth and latency requirements of other VNFs should be guaranteed. Furthermore, the type of VNF determines the value of k_p, which is closely related to the service rate of VNF and allocated resources. Given a certain k_p, the bandwidth requirement b_i for SFC s_i determines the lower bound for allocated resource $r_{\min}(f)$. The difference in packet arrival rates and service rate can be used to evaluate the degree of packet loss, i.e., $\lambda - \mu$. This value should be minimized to reduce packet loss.

4.2. Resource scheduling

The main purpose for the phase of resource scheduling is to determine the appropriate amount of resources to allocate if a VNF is to be deployed on an NFV server. Therefore, the overall cost of packet loss should be minimized. Moreover, if VNFs of multiple SFCs are deployed on the same server, their bandwidth requirements must be satisfied.

The resource scheduling algorithm of *Palos* is shown in Algorithm 1. It determines the appropriate resource allocation on server v in which a

new VNF $f_{n,m}$ is to be deployed. When there is no VNF deployed on server v, the resource of server v will be evaluated first (line 2 ~6). If resource is insufficient to satisfy the bandwidth requirement of the SFC, the algorithm will return a signal of insufficient resource (line 3 ~5). Otherwise, the appropriate amount of resource $\frac{\lambda_{n,m}}{k_p}$ that can completely serve the packet arrival rate $\lambda_{n,m}$ for VNF $f_{n,m}$ will be calculated. This should not exceed $r_{\max}(p)$, which is the corresponding resource for the maximum service rate of the VNF. Then, the VNF $f_{n,m}$ will be allocated the amount of resource either $\frac{\lambda_{n,m}}{k_p}$ or $C(v)$, whichever is smaller (line 6).

Algorithm 1 *Palos*: Resource Scheduling

Input: $G = (V, E)$, network; S, a set of SFCs; v, target server; $f_{n,m}(p)$, VNF of type p.
Output: Resource allocation R on server v after deploying a new VNF $f_{n,m}(p)$.

1: $R \leftarrow dict()$ ▷ Initialized as a dictionary
2: **if** v contains no VNFs **then**
3: **if** $C(v) < \frac{b_n}{k_p}$ **then**
4: **return** ▷ No sufficient resource
5: **end if**
6: $R[f_{n,m}] \leftarrow min(C(v), \frac{\lambda_{n,m}}{k_p}, r_{max}(p))$
7: **else**
8: $C(v)' \leftarrow C(v) - \sum_{f_{i,j} \in v} \frac{b_i}{k_p}$
9: **if** $C(v)' < \frac{b_n}{k_p}$ **then**
10: **return** ▷ No sufficient resource
11: **else**
12: Pre-deploy $f_{n,m}$ to v
13: $W \leftarrow dict()$ ▷ Initialized as a dictionary
14: **for** each $f_{i,j}$ on v **do**
15: $Q_{i,j} \leftarrow \sum_{h<j} \frac{\lambda'_{i,h}}{r_{i,h} \cdot k_p}$
16: $W[f_{i,j}] \leftarrow \lambda_{i,j} \cdot Q_{i,j}$
17: **end for**
18: $Sum_v \leftarrow \sum_{f_{i,j} \in v} W[f_{i,j}]$
19: **for** each $f_{i,j}$ on v **do**
20: $R[f_{i,j}] \leftarrow \frac{b_i}{k_p} + C(v)' \cdot \frac{W[f_{i,j}]}{Sum_v}$
21: **end for**
22: **end if**
23: **end if**
24: **return** R

When there are one or more VNFs already instantiated on the server, the decisions of resource scheduling will be complicated. Not all VNFs can achieve the maximum service rate due to limited resources on the server. To minimize packet dropping cost, *Palos* comprehensively considers packet arrival rate λ and position weights of each VNF in server v. The position weight of $f_{i,j}$ in v is defined in Equation (12).

$$Q_{i,j} = \sum_{h<j} \frac{\lambda'_{i,h}}{k_p \cdot r_{i,h}} \tag{12}$$

$Q_{i,j}$ is the sum of the ratio of effective packet arrival rate to service rate for upstream VNFs of $f_{i,j}$. According to queuing theory, this value can be used to evaluate the busyness of a single-service system. $Q_{i,j}$ can reflect not only the relative position of the VNF in the current SFC, but also the degree of resources consumed by upstream VNFs. It is obvious that the system load is proportional to packet arrival rate and inversely proportional to service rate. Moreover, not all VNFs deployed on the same server will obtain more resources if they are in a later position in the SFC. For example, VNF1 is the fifth VNF of one SFC while VNF2 is the third VNF of another SFC and both are deployed on the same server. When the overall load of upstream VNFs for VNF2 is relatively higher, the packet dropping cost caused by VNF2 may be higher. Therefore, it will be unfair if more resources are allocated to VNF1 simply because VNF1 is in a later position in the SFC than VNF2.

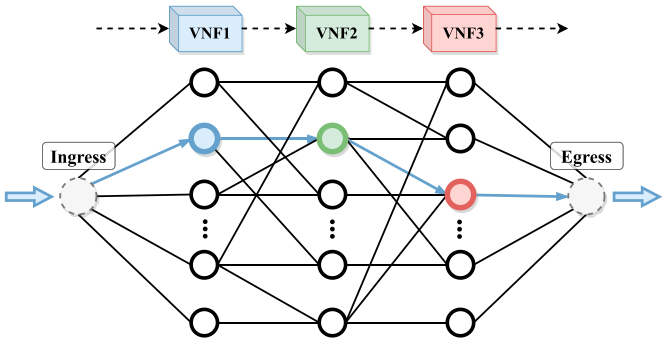


Fig. 5. *Palos* SFC deployment.

Based on the above analysis, for each VNF that is already deployed on the server, the minimum amount of resource (i.e., $\frac{b_i}{k_p}$) required to ensure the bandwidth requirement of the SFC are calculated (line 8). Then, if residual resource $C(v)'$ of the server is less than the minimum resource requirement $\frac{b_n}{k_p}$ of the VNF to be deployed, a signal of insufficient resource will be returned (line 9 ~10). Otherwise, position weights of all VNFs (including $f_{n,m}$ to be deployed) are calculated (line 15). The product of $Q_{i,j}$ and $\lambda_{i,j}$ shows that consumption degree of upstream resources is related to the arrival rate to $f_{i,j}$. High packet loss will cause resource wastage even if $Q_{i,j}$ is minimal. After determining their position weights, resources are scheduled for multiple VNFs on the same server (line 19 ~21). Under the premise of ensuring the minimum resource $\frac{b_i}{k_p}$, each VNF on the server will be allocated as much resource as possible according to its weight (line 20).

Algorithm 1 either allocates resources to the VNF to be deployed (line 6), or adjusts resources for those deployed VNFs on the server v (line 19 ~21). It is easily observed that time complexity for the phase of resource scheduling is $\mathcal{O}(A)$, where A is the number of VNFs that needs to be deployed on the server v.

4.3. Deployment of SFCs

The SFC deployment phase will determine positions and paths for VNFs, as well as their resource allocation based on Section 4.2. The goal is to minimize packet dropping cost while ensuring both the bandwidth and latency requirements of SFC paths. Given a network topology and users' requests for SFCs, the procedures of SFC deployment are as follows: (i) suitable servers with limited computing resources are chosen for instantiation of VNFs; (ii) resource is appropriately allocated to VNFs; (iii) the best paths for data flow processed by SFCs are calculated; and (iv) placement of SFCs needs to consider SFC performance and SLAs requirements.

To deploy SFCs, a strategy based on dynamic programming is proposed, and the Viterbi algorithm [20] is introduced to improve performance. The Viterbi algorithm is widely used in solving hidden Markov and conditional random field prediction. As shown in Fig. 5, for each SFC,

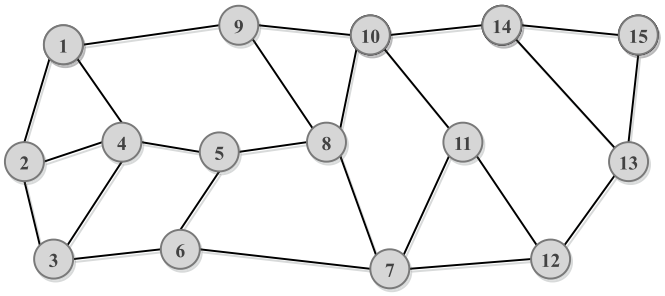


Fig. 6. Internet2 with advanced layer 3 service.

a corresponding VNF-to-Server diagram is generated according to the physical network topology, where columns are called layers of the SFC and each row represents the same server. The diagram has two virtual nodes, i.e., the *ingress* and *egress* of an SFC. The edges correspond to the physical paths of the network. Suppose the length of the SFC is J and there are $|V|$ servers in the physical topology, a VNF-to-Server diagram with size of $|V| \times J$ can be generated.

Algorithm 2 *Palos*: SFC Deployment

Input: $G = (V, E)$, network; S, a set of SFCs.
Output: Path and position of $s_i \in S$.

1: **for** $s_i \in S$ **do**
2: Initialize $G' \leftarrow G$
3: **for** each layer of $f_{i,j}$ in s_i **do**
4: $Z \leftarrow \varnothing$
5: **for** each node v in current layer **do**
6: **if** $\alpha \le w_i$ and $\varphi \ge b_i$ on path (u, v) **then**
7: $Weight_{j,v,u} \leftarrow 0$
8: Schedule resource using Algorithm 1
9: **if** no sufficient resource **then**
10: **break**
11: **end if**
12: $L(f) \leftarrow$ Compute the number of dropped packets by Equation (9)
13: **for** each path (u, v) **do**
14: $Weight_{j,v,u} \leftarrow L(f) \cdot (\sum \alpha + \sum \beta)$
15: **end for**
16: $Z \leftarrow Z \cup \{v\}$
17: **end if**
18: **end for**
19: Record *Position* by Z
20: $Cost_{j,v,u} \leftarrow 0$
21: **for** $v \in Z$ **do**
22: $Cost_{j,v,u} \leftarrow min(\sum_{h \le j} Weight_{j,v,u})$
23: $\varphi(u, v) \leftarrow \varphi(u, v) - b_i$
24: Retain paths according to $Cost_{j,v,u}$
25: **end for**
26: **end for**
27: Select path to *egress* with $\alpha \le w_i, \varphi \ge b_i$ and $min(Cost_{j,v,u})$ of last layer
28: $Path \leftarrow$ Backward tracking of paths
29: **end for**
30: **return** *Path* and *Position* for each s_i deployment

Starting from the first VNF of the SFC, the algorithm chooses a server from layer one that can satisfy both bandwidth and latency requirements, and then allocates resources to the VNF according to Algorithm 1. Afterwards, the number of dropped packets during a time period ΔT is calculated, and then the weights of paths connected to the server are obtained (line 12 ~15). Algorithm 2 applies the Viterbi algorithm to find the shortest path, i.e., each node in layer two and onward only keeps information of the shortest path connected by it. The Viterbi algorithm requires that at each layer, the non-shortest path connecting the same node is pruned. Compared to traversing all paths, this method reduces the time complexity from $\mathcal{O}(|V|^J)$ to $\mathcal{O}(J \cdot |V|^2)$.

On the other hand, the bandwidth of each path will change according to the status of deployed SFCs. The propagation delay of all edges leaving *ingress* and entering *egress* are set to 0.

Algorithm 2 initializes the graph and the data structure storing graph information (line 2). Then, for each layer of the VNF-to-Server diagram, it calculates the path weights for the appropriate paths connecting the nodes of the current layer (line 5 ~18). It prunes the paths and retains the paths and nodes information. After the calculation of each VNF layer of the current SFC is completed, the best path is obtained by backward tracking (line 21 ~25).

The most time-consuming part of the algorithm is calculating the shortest path for each SFC. Algorithm 2 uses the Viterbi algorithm to reduce the time complexity of that part to $\mathcal{O}(J \cdot |V|^2)$. Therefore, the time complexity for deploying $|S|$ SFCs is $\mathcal{O}(|S| \cdot J \cdot |V|^2)$.

5. Performance evaluation

5.1. Experiment setup

Network topology. The network topology used in experiments is based on Internet2[6], which contains 15 nodes and 24 physical links, as shown in Fig. 6. The bandwidth of each link is 40 Gbps. Link latency of each link is 5 ~20 ms. The capacity of each server scales from 20% to 80%.

VNFs & SFCs. Table 4 shows five types of VNFs. These VNFs are chosen randomly to form different SFCs each of which contain three VNFs. The ingress and egress of each SFC are stochastically selected. According to SLAs, the minimum throughput requirement for each SFC is limited to 300–600 *pps*, delay is limited to 15–30 ms, and the arrival rate of packets to SFCs is limited to 600–1200 *pps*. ΔT is set to 10 ms.

Baseline algorithms. Two baseline algorithms, i.e., NFO-DP (Dynamic Programming-based Network Function Orchestration) and TS-NFMS (Tabu Search-based Network Function Mapping and Scheduling), are used for comparisons. NFO-DP [9] aims to minimize deployment and data forwarding cost by using the Viterbi method to deploy SFCs. Furthermore, it uses a round-robin scheduler to allocate resources for each node. TS-NFMS [21] applies Tabu to place VNFs using greedy strategy and searches local servers for SFC deployment, which simplifies the selection of appropriate servers.

5.2. Results and analysis

Packet loss rate. Fig. 7 shows average packet dropping rates on VNFs at different positions for the three algorithms. The number of SFCs is set to 10. It is shown that *Palos* outperforms the other two algorithms in the aspect of packet dropping rates wherein the average packet loss rates of the last two VNFs are only 0.668% and 0.524%. *Palos* has an overview of relative positions of currently deployed VNFs as well as the states of other VNFs to reasonably allocate resources. Packet loss rate of the first VNF is a bit higher because it is placed near ingress servers, where most VNFs are deployed such that resources are quickly consumed. Resource contention among these VNFs intensifies packet dropping. NFO-DP uses a round-robin scheduler to allocate resources to VNFs, which leads to unfairness in resource allocation when the number of SFCs to be deployed is large. This further causes an overall high packet dropping rate, i.e., 5.268% for the second VNF. TS-NFMS has no obvious advantages in reducing packet loss rate due to its inability to sense relative positions of VNFs and traffic flows.

Packet dropping cost. In this experiment, average packet dropping cost is evaluated under different numbers of SFCs (i.e., 2 ~16). As shown in Fig. 8, *Palos* can minimize packet dropping cost. Compared to NFO-DP and TS-NFMS, *Palos* improves packet dropping cost by up to 42.73% and 29.05%, respectively. Note that NFO-DP has a lower packet dropping cost than *Palos* when the number of SFCs is between 2 and 8. This is because few VNFs are deployed on one or more servers so that resources are abundant for these VNFs to process traffic flows in this situation. However NFO-DP treats resource allocation to VNFs equally, which causes the packet dropping cost rate to increase rapidly as the number of SFCs increases. When the number of SFCs changes from 4 to 8, the TS-NFMS packet dropping cost rate also increases sharply. The reason is that NFV servers are saturated when the number of SFCs is greater than 4. In this circumstance, packet loss rate increases because of lack of resources. As more SFCs are deployed, packet loss rate stabilizes because TS-NFMS chooses nearby idle servers to instantiate VNFs.

[6] https://internet2.edu/.

Table 4
Deployed VNF collections.

VNF type	μ_{max} (*pps*)	r_{max} (%)	$N(f)$ (#)
VNF1	800	1	200
VNF2	800	2.5	500
VNF3	1000	2	800
VNF4	700	5	800
VNF5	600	4	900

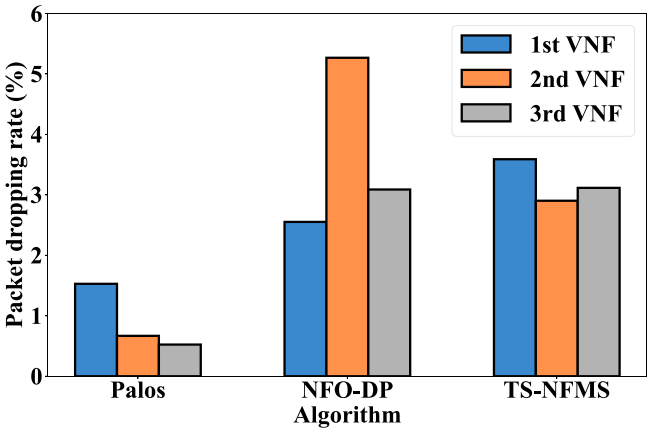


Fig. 7. Average packet dropping rate at different positions of the SFC.

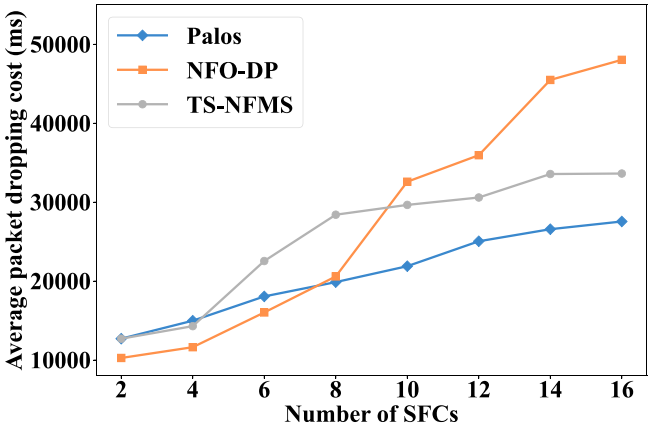


Fig. 8. Average packet dropping cost varying along with the number of SFCs.

SFC latency. Average latency of SFCs is also evaluated. Fig. 9 shows that *Palos* does not incur any extra latency when trying to deploy SFCs with an objective to minimize packet dropping cost. That is because *Palos* constrains the minimum bandwidth of SFCs and senses the loads of upstream VNFs in the SFCs. Furthermore, the mechanism of *Palos* assures the constraint of latency, which includes propagation delay, queuing latency and processing latency. When the number of SFCs is small, the latency that *Palos* incurs is slightly higher than the other two algorithms. The main reason is that *Palos* inclines to acquire more resources from different servers, which introduces more latency. NFO-DP has advantages of reducing latency in the case where there are few VNFs. However, it causes severe packet loss when the number of VNFs increases, which in return incurs a higher latency. When the number of SFCs ranges from 2 to 6, TS-NFMS can well optimize SFC latency as it reduces link latency as much as possible while there are abundant resources for SFCs to handle traffic flows. Compared to NFO-DP and TS-NFMS, *Palos* can reduce average SFC latency by up to 33.03% and 28.64%, respectively, when the number of SFCs is 16.

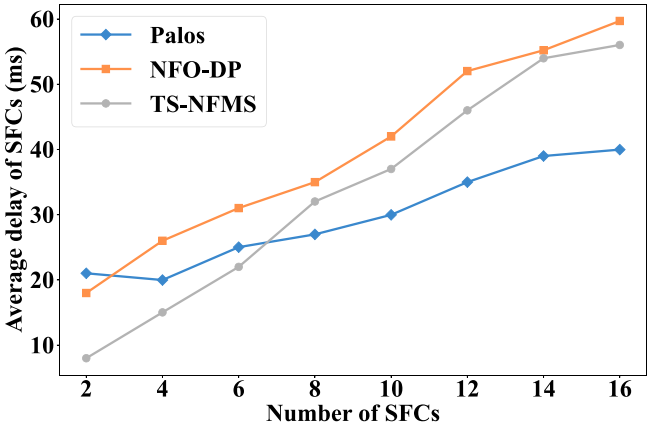


Fig. 9. Average delay varying along with the number of SFCs.

Per SFC performance. As seen from Figs. 8 and 9, when more SFCs are deployed, the corresponding average packet dropping cost and average delay increase as well. Both figures reflect that the number of deployed SFCs is an important factor that affects the performance of each algorithm. Moreover, SFC performance can also be evaluated among three algorithms as shown in Figs. 10 and 11. The average packet dropping cost of an SFC by *Palos* is less than 3000 ms, demonstrating that *Palos* can effectively reduce packet dropping costs compared with the other two algorithms. However, the average delay of an SFC by *Palos* is slightly higher than that of TS-NFMS. This is reasonable as the average delay of SFCs is much lower in the case of TS-NFMS when the number of SFCs is small. Nevertheless, the delay with around 4 ms is acceptable in most cases of deployed SFCs. Furthermore, both figures show the Rate Of Change (ROC) of average packet dropping cost and average delay when an increasing number of SFCs are deployed. The ROCs for *Palos* are much lower (~10%) than those of the other two algorithms, indicating slower growth in average packet dropping cost and average delay even if many SFCs are to be deployed.

CPU utilization. Furthermore, an experiment based on CPU utilization of non-idle servers is conducted with 10 deployed SFCs. Fig. 12 demonstrates that *Palos* does not achieve the maximum utilization of CPU compared to NFO-DP and TS-NFMS. To minimize packet dropping cost, *Palos* tries to deploy SFCs on adjacent servers with abundant resources. However, it does not fully consider the number of chosen servers. In other words, instead of saving CPU resources, *Palos* is more likely to use sufficient resources to reduce overhead. NFO-DP will take into account the minimization of deployment and data forwarding cost. Therefore, it concentrates more on the selection of servers. Furthermore, TS-NFMS adopts an integrated strategy for the deployment of VNFs, aggressively deploying VNFs on servers with maximum available resources. In addition, when failing to deploy VNFs on the current server, TS-NFMS uses Tabu-search [21] to find the optimal nearby server to carry the deployment. Hence, it achieves maximum CPU utilization compared to the other two algorithms.

6. Related works

6.1. SFC resource scheduling

Many works focus on resource scheduling of SFCs to optimize resource utilization. Qu et al. [22] studied the impacts of transmission and processing delays of VNFs on resource scheduling. They formulated the problem as an MILP to be solved using a genetic algorithm-based method. Li and Cui [10] focused on position-aware resource scheduling on a single server and the cost caused by dropping packets which have been processed by upstream VNFs in the SFC. The proposed Minimizing Dropping Cost (MDC) algorithm can achieve differentiated resource

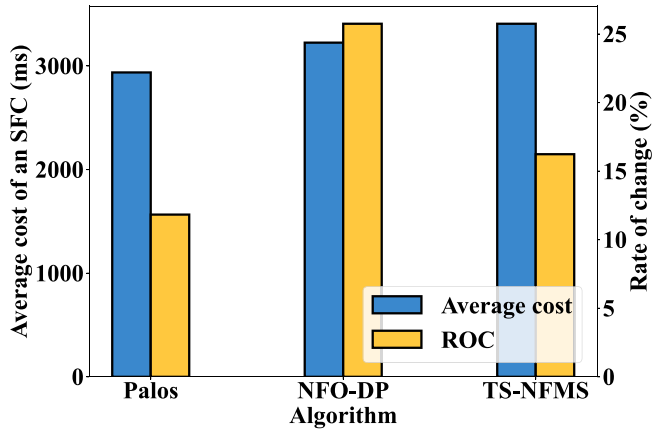


Fig. 10. Average packet dropping cost per SFC in different scenarios.

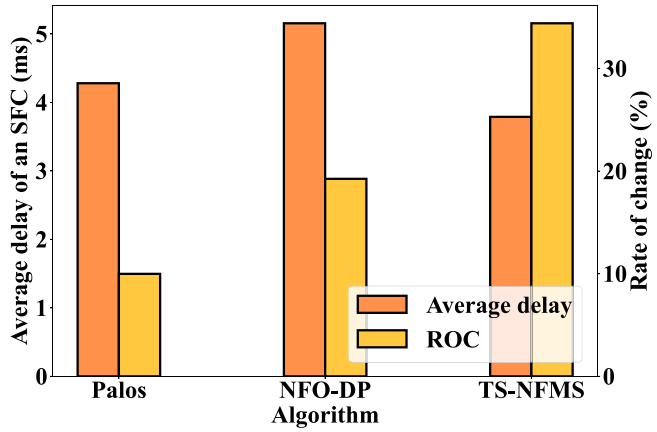


Fig. 11. Average delay per SFC in different scenarios.

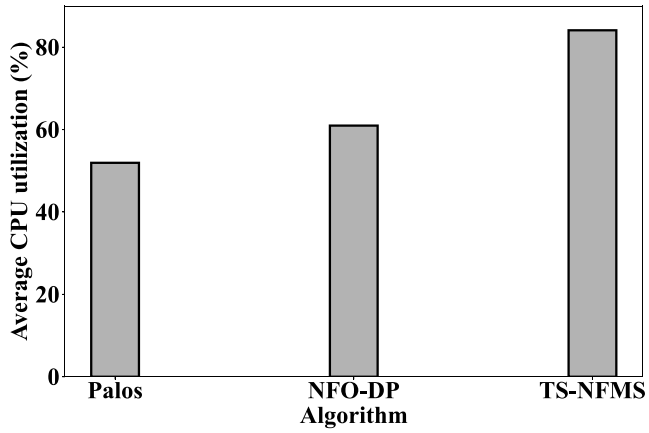


Fig. 12. Average CPU utilization with 10 SFCs.

allocation for VNFs at different positions in the SFC. Anthony et al. [23] presented a User-space Non-intrusive work-flow aware VNF Scheduler (UNiS) to allocate CPU resources to poll-mode VNFs. They also considered SFC processing order (i.e., position-aware) when scheduling VNFs. Yao et al. [24] designed an efficient mapping and preemptive scheduling of functions for multiple service chains to minimize completion time of the whole system. Pham et al. [25] proposed a matching-based algorithm to solve the NP-hard VNF scheduling problem in which VNFs and resource slots are cast as players in a one-to-one matching game. By allowing resource sharing and preemption, Zhang et al. [26] studied VNF instances deployed on the same node to share the computational resources. They aimed to optimize the acceptance ratio of arriving network services. In addition, there are also some works focusing on intelligent offloading policy, e.g. Refs. [27–29].

6.2. SFC deployment

Jiang et al. [4] formulated the SFC placement problem as an MILP model which aims to minimize the end-to-end packet loss rate of an SFC. They proposed an Ant Colony System (ACS) based algorithm to solve the placement problem considering bandwidth and delay constraint. Using machine learning techniques, Subramanya et al. [5] employed Integer Linear Programming (ILP) to formulate and solve a joint user association and SFC placement problem in the environment of Multi-access Edge Computing (MEC). They aimed to minimize the overall end-to-end latency. To reduce deployment cost and service latency with respect to a variety of constraints, Khoshkholghi et al. [6] formulated a multi-objective optimization model to joint VNF placement and link embedding using two heuristic-based algorithms to optimize their objectives. The work [30] aimed to minimize end-to-end latency by formulating and solving a joint user association, SFC placement and resource allocation problem in the 5G network. Liu et al. [31] studied dynamical VNF placement and routing problem for SFC request flows. They aimed to optimize throughput and average traffic acceptance rate. Wang et al. [32] studied the parallelized SFC placement problem in data center networks with an objective to optimize SFC delay and link consumption while guaranteeing availability.

Above works mainly aim to optimize resource consumption, latency and deployment cost by scheduling SFCs. Different from existing works, this study explores the novel idea that packet dropping cost at different positions in SFCs are different for SFC placements.

7. Conclusions

This study investigated resource scheduling on SFC deployment for minimizing total packet dropping cost. We found that when SFCs are deployed on servers, appropriate resource scheduling can not only maximize the performance of SFCs but also reduce resource wastage. The proposed solution *Palos* is a scheme for scheduling resources and deployment with sensitivity to positions and arrival rates. Retransmitted packets are treated as new packets of the flow because packet retransmission is usually handled by user applications. In a future study, further impacts and optimization for packet retransmission will be considered.

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.

Acknowledgments

This work has been partially supported by the National Natural Science Foundation of China (NSFC) No. 62172189 and 61772235; the Natural Science Foundation of Guangdong Province No. 2020A1515010771; the Science and Technology Program of Guangzhou No. 202002030372; the UK Engineering and Physical Sciences Research Council (EPSRC) grants EP/P004407/2 and EP/P004024/1, and Innovate UK grant 106199–47198.

References

[1] L. Askari, M. Tamizi, O. Ayoub, M. Tornatore, Protection strategies for dynamic VNF placement and service chaining, in: International Conference on Computer Communications and Networks (ICCCN), IEEE, 2021, pp. 1–9.

[2] J. Sun, F. Liu, H. Wang, M. Ahmed, Y. Li, M. Liu, Efficient VNF placement for Poisson arrived traffic, IEEE Trans. Network. Service Manage. 18 (2021) 4277–4293.

[3] W. Liang, L. Cui, F.P. Tso, Low-latency service function chain migration in edge-core networks based on open Jackson networks, J. Syst. Architect. 124 (2022) 102405.

[4] Y. Jiang, X. Wang, T. Zhao, Y. Wang, P. Yu, Deployment algorithm of service function chain with packet loss rate optimization, in: Proceedings of the 9th International Conference on Computer Engineering and Networks, Springer, 2021, pp. 635–645.

[5] T. Subramanya, D. Harutyunyan, R. Riggio, Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks, Comput. Network. 166 (2020) 106980.

[6] M.A. Khoshkholghi, M.G. Khan, K.A. Noghani, J. Taheri, D. Bhamare, A. Kassler, Z. Xiang, S. Deng, X. Yang, Service function chain placement for joint cost and latency optimization, Mobile Network. Appl. 25 (2020) 2191–2205.

[7] W. Hajji, T.A. Genez, F.P. Tso, L. Cui, I. Phillips, Dynamic network function chain composition for mitigating network latency, in: IEEE Symposium on Computers and Communications (ISCC), IEEE, 2018, pp. 316–321.

[8] L. Cui, F.P. Tso, S. Guo, W. Jia, K. Wei, W. Zhao, Enabling heterogeneous network function chaining, IEEE Trans. Parallel Distr. Syst. 30 (2019) 842–854.

[9] M.F. Bari, S.R. Chowdhury, R. Ahmed, R. Boutaba, On orchestrating virtual network functions, in: The 11th International Conference on Network and Service Management (CNSM), IEEE, 2015, pp. 50–56.

[10] C. Li, L. Cui, A novel NFV schedule optimization approach with sensitivity to packets dropping positions, in: Proceedings of the Workshop on Theory and Practice for Integrated Cloud, Fog and Edge Computing Paradigms, 2018, pp. 23–28.

[11] S.G. Kulkarni, W. Zhang, J. Hwang, S. Rajagopalan, K. Ramakrishnan, T. Wood, M. Arumaithurai, X. Fu, NFVnice: dynamic backpressure and scheduling for NFV service chains, IEEE/ACM Trans. Netw. 28 (2020) 639–652.

[12] C. Pham, N.H. Tran, S. Ren, W. Saad, C.S. Hong, Traffic-aware and energy-efficient vNF placement for service chaining: joint sampling and matching approach, IEEE Trans. Service Comput. 13 (2017) 172–185.

[13] S. Ahvar, M.M. Mirzaei, J. Leguay, E. Ahvar, A.M. Medhat, N. Crespi, R. Glitho, SET: a simple and effective technique to improve cost efficiency of VNF placement and chaining algorithms for network service provisioning, in: The 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), IEEE, 2018, pp. 293–297.

[14] F. Schardong, I. Nunes, A. Schaeffer-Filho, A distributed NFV orchestrator based on BDI reasoning, in: IFIP/IEEE Symposium on Integrated Network and Service Management (IM), IEEE, 2017, pp. 107–115.

[15] Q. Zhang, Y. Xiao, F. Liu, J.C. Lui, J. Guo, T. Wang, Joint optimization of chain placement and request scheduling for network function virtualization, in: IEEE 37th International Conference on Distributed Computing Systems (ICDCS), IEEE, 2017, pp. 731–741.

[16] T. Wang, F. Liu, J. Guo, H. Xu, Dynamic SDN controller assignment in data center networks: stable matching with transfers, in: The 35th Annual IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.

[17] X. Li, C. Qian, Low-complexity multi-resource packet scheduling for network function virtualization, in: IEEE Conference on Computer Communications (INFOCOM), IEEE, 2015, pp. 1400–1408.

[18] J.F. Shortle, J.M. Thompson, D. Gross, C.M. Harris, Fundamentals of Queueing Theory, vol. 399, John Wiley & Sons, 2018, https://doi.org/10.1002/9781119453765.

[19] C. Chekuri, S. Khanna, A polynomial time approximation scheme for the multiple knapsack problem, SIAM J. Comput. 35 (2005) 713–728.

[20] G.D. Forney, The Viterbi algorithm, Proc. IEEE 61 (1973) 268–278.

[21] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, S. Davy, Design and evaluation of algorithms for mapping and scheduling of virtual network functions, in: Proceedings of the 1st IEEE Conference on Network Softwarization (NetSoft), IEEE, 2015, pp. 1–9.

[22] L. Qu, C. Assi, K. Shaban, Delay-aware scheduling and resource optimization with network function virtualization, IEEE Trans. Commun. 64 (2016) 3746–3758.

[23] A. Anthony, S.R. Chowdhury, T. Bai, R. Boutaba, J. Francois, Non-intrusive and workflow-aware virtual network function scheduling in user-space, IEEE Trans. Cloud. Comput. (2020), https://doi.org/10.1109/TCC.2020.3024232, 1–1.

[24] H. Yao, M. Xiong, H. Li, L. Gu, D. Zeng, Joint optimization of function mapping and preemptive scheduling for service chains in network function virtualization, Future Generat. Comput. Syst. 108 (2020) 1112–1118.

[25] C. Pham, N.H. Tran, C.S. Hong, Virtual network function scheduling: a matching game approach, IEEE Commun. Lett. 22 (2017) 69–72.

[26] Y. Zhang, F. He, T. Sato, E. Oki, Optimization of network service scheduling with resource sharing and preemption, in: IEEE 20th International Conference on High Performance Switching and Routing (HPSR), IEEE, 2019, pp. 1–6.

[27] L. Zhang, B. Cao, Y. Li, M. Peng, G. Feng, A multi-stage stochastic programming-based offloading policy for fog enabled iot-ehealth, IEEE J. Sel. Area. Commun. 39 (2021) 411–425.

[28] Y. Li, S. Xia, M. Zheng, B. Cao, Q. Liu, Lyapunov optimization-based trade-off policy for mobile cloud offloading in heterogeneous wireless networks, IEEE Trans. Cloud. Comput. 10 (2022) 491–505.

[29] B. Cao, L. Zhang, Y. Li, D. Feng, W. Cao, Intelligent offloading in multi-access edge computing: a state-of-the-art review and framework, IEEE Commun. Mag. 57 (2019) 56–62.

[30] D. Harutyunyan, N. Shahriar, R. Boutaba, R. Riggio, Latency and mobility-aware service function chain placement in 5G networks, IEEE Trans. Mobile Comput. (2020), https://doi.org/10.1109/TMC.2020.3028216, 1–1.

[31] L. Liu, S. Guo, G. Liu, Y. Yang, Joint dynamical VNF placement and SFC routing in NFV-enabled SDNs, IEEE Trans. Network. Service Manage. 18 (2021) 4263–4276.

[32] M. Wang, B. Cheng, S. Wang, J. Chen, Availability- and traffic-aware placement of parallelized SFC in data center networks, IEEE Trans. Network. Service Manage. 18 (2021) 182–194.