Mobile Networks and Applications
https://doi.org/10.1007/s11036-021-01868-5

JOSP: Joint Optimization of Flow Path Scheduling and Virtual Network Function Placement for Delay-Sensitive Applications

Qing Lyu[1] · Yonghang Zhou[2] · Qilin Fan[3] · Yongqiang Lyu[4] · Xi Zheng[5] · Guangquan Xu[6] · Jun Li[1]

Accepted: 27 September 2021

Abstract
With the rapid development of network function virtualization, delay-sensitive applications including auto-driving, online gaming, and multimedia conferencing can be served by virtual network function (VNF) chains with low operation expense/capital expense and high flexibility. However, as the service requests are highly dynamic and different services require distinct bandwidth occupation amount and time, how to schedule the paths of flows and place VNFs efficiently to guarantee the performances of network applications and maximize the utilization of the underlying network is a challenging problem. In this paper, we present a joint optimization approach of flow path scheduling and VNF placement, named JOSP, which explores the best utilization of bandwidth from two different aspects to reduce the network delay. We first present a delay scaling strategy that adds the penalty to the link bandwidth occupation that may cause congestion in accordance with the network placement locations. Then we consider the bandwidth occupation time and present a long-short flow differentiating strategy for the data flows with different duration. Furthermore, we present a reinforcement learning framework and use both the flow path delay and the network function-related delay to calculate the reward of placing VNFs adaptively. Performance evaluation results show that the JOSP can reduce the network delay by 40% on average compared with the existing methods.

Keywords Service chain · Flow path scheduling · VNF placement · Joint optimization · Reinforcement learning

1 Introduction

The emerging network technologies such as 5G [1] and Internet of Things [2] have promoted a wide range of delay-sensitive applications, including auto-driving, online gaming, multimedia conferencing, and virtual reality [3–5].These network applications not only need to pass through a series of specific network functions (e.g., load balancing, traffic monitoring, caching, deep inspection) [6] but also have to fulfill the stringent requirement of response time [7]. With the advances of network function virtualization (NFV), network functions can be executed by software middle-boxes through virtualization rather than dedicated hardware, known as virtualized network functions (VNFs).In these scenarios, the placement of VNFs should be carefully decided and the flow path should be carefully scheduled to guarantee the performances of network applications (i.e., reducing the network delay) [8, 9] and maximize the utilization of underlying network.

Previous works have made various efforts to address this issue [6, 9–17]. Some works focused on the flow path scheduling to find the shortest paths for flows by exploring the diversity of network topology [9, 11]. Other works focused on the VNF placement problem and formulated it as the integer linear programming problem. Then, these works proved the problem to be NP-hard, and used heuristic methods [6, 18] or machine learning-based methods [15, 16] to solve it. However, existing works still face the following challenges: i) As different occupation amount and time of the network bandwidth is required by different types of flows under the limited bandwidth capacity, the bandwidth allocation of an incoming flow would influence the fulfilment of the subsequent flows [12]. Therefore, The influence of bandwidth occupation amount and time should be further explored to improve the network utilization; ii) Learning-based methods only considered the flow path delay in the reward calculation while ignoring the function-related delay which is also a significant indicator for

✉ Qilin Fan
fanqilin@cqu.edu.cn

Extended author information available on the last page of the article.

Published online: 04 February 2022

deciding the placement locations; iii) The VNF placement has an impact on the path scheduling as the flow needs to traverse to the locations of VNFs to enforce the service chain on the path. Likewise, in the delay-aware scenario, the network function placement needs to consider the delay generated by the flow traversing. Therefore, the flow path scheduling has an impact on the placement. However, existing works neglected the impact of the path scheduling and VNF placement on each other.

Improper flow path scheduling and VNF placement scheme would exacerbate the unbalanced utilization of network bandwidth, generate the congestion [13] and increase the delay [10]. Therefore, the flow path scheduling and VNF placement should be jointly considered to satisfy the requirements of flows under different network states.

In this paper, we develop a framework named Joint Optimization of flow path Scheduling and virtual network function Placement (JOSP) to reduce network delay. Unlike traditional works that consider the flow path scheduling and VNF placement separately, JOSP combines them together as these two problems are correlated with each other.

In order to avoid the delay increased by the unreasonable bandwidth occupation, JOSP presents two strategies, the DElay Scaling (DES) and the Long-Short Flow Differentiating (LSFD). The DES and LSFD explore the best utilization of network bandwidth in flow path scheduling. Empowered by the DES and LSFD strategies, JOSP adopts a reinforcement learning method to determine the placement locations of VNFs.

The main contributions of the paper are listed as follows:

- We investigate the problem of flow path scheduling and VNF placement jointly and formulate it as an integer linear programming to minimize the delay. Both the path scheduling constraints and the function placement constraints are considered.
- We present two strategies to reduce the total end-to-end delay in the path scheduling. Particularly, we analyze the link occupation situation and present a delay penalty mechanism to overcome the unreasonable occupation a-mount of the bandwidth. We analyze the distribution characteristics of the flow duration and present a scheduling algorithm to differentiate flows with distinct bandwidth occupation time.
- We propose a reinforcement learning (RL) based method to obtain the optimal locations of VNFs. JOSP takes both the flow path delay and network function-related delay to calculate the reward to assist improving the VNF placement.
- We conduct extensive experiments on JOSP based on the real-world topologies. the results show that JOSP can reduce the network delay by 40% on average compared to existing methods.

The remainder of this paper is organized as follows. we introduce the related work of flow path scheduling and VNF placement in Section 2. The network model and problem formulation are given in Section 3. The detailed design of JOSP is presented in Section 4. Simulation results validate the effectiveness of the JOSP in Section 5. Finally, we conclude this paper and discuss the future work in Section 6.

2 Related Work

Flow path scheduling and VNF placement have been studied in intensively in recent years. However, most of the existing work dealt with the problems independently.

2.1 Flow Path Scheduling

A flow needs to pass through a set of VNFs in sequence, known as the service chain. So, the path of flows should be scheduled in accordance with the locations of VNFs. Besides, as the bandwidth of the network is limited, not all flows can be scheduled to desired paths. Therefore, the path of flows should be scheduled to conform to the network status to avoid unnecessary congestion and reduce the delay [19, 20]. Many efforts have been put on solving the low path scheduling problem from different angles.

A topology triggered method to find low-delay path for flows was put forward in [9]. It explored the diversities of different paths in the network and introduced a metric called low-latency path diversity to present the capacity of topology. It was measured to find the low-delay paths on the demand of no congestion. Empirical data [21] was used to improve the network utilization and availability to reduce the failures from links or nodes. It provided multi-cast service for service chain related traffic scheduling to reduce delay. A method called fast path assignment [22] was provided to minimize the influence of the bandwidth occupation of the bottleneck links in the scheduling of flow paths, but the overall bandwidth utilization was not considered. Other works considered the energy consumption [23], flows scheduling in different sub-networks [14], placement reliability as well as CPU utilization in 5G scenario [1, 2] when scheduling traffic through VNFs.

However, as the bandwidth is limited, the bandwidth occupation of a flow would affect the path scheduling of other flows. The influence of the bandwidth occupation amount and time of the current flow on subsequent flows were not considered in the path scheduling in these works.

2.2 VNF Placement

Many works formulated VNF placement into the programming problems, such as mixed-integer linear programming

or non-linear integer programming, and designed heuristics to solve the problem [13, 15]. In different application scenarios, different optimization objectives were considered, such as maximizing link utilization [18], minimizing network delay [13, 24], minimizing response time under the constraint of set up cost [15], improving resource usage with time-varying workload [25], and minimizing the bandwidth consumption [6]. The feasibility of the placement and management of VNFs in software-defined network (SDN) circumstance was studied in [26]. It presented the industry concerns of resource management and policy composition problem. These works provided fine-tuned heuristic information-based solutions to the VNF placement. Most of the solutions began from a greedy manner and aimed at finding the near-optimal placement locations of VNFs. For example, solving the VNF placement by greedy method and further adjusting the placement locations in the next stage by using simulated annealing method [6]. These methods relied on the tuning of heuristics and could not be scaled to other scenarios when the network environment changes.

Other works proposed algorithms based on machine learning (ML) to address the VNF placement issue. For example, a two-phase ML-based algorithm [8] was introduced to minimize the network delay, which used two types of deep belief networks (for function selection and service chaining separately) to optimize the placement. A decision tree model was presented to decide the placement of VNFs in [27], then it took features related performance as the input of the learning framework to improve the placement. The attempts of adopting ML methods to resolve the placement problem are encouraging, but the raw data collected from network would lead to slow learning convergence and limited performance improvement.There were also works using the RL based methods to solve the VNF placement problem [16, 28], where the reward calculation should be carefully considered.

However, these works only used flow path delay to calculate the reward, ignoring the network function-related delay. The network function-related delay indicates the influence of the placement location of the VNF on the delay. Therefore, it should be sufficiently explored to improve the placement performance.

Our approach in this paper is differentiated by solving the flow path scheduling and VNF placement problems all together. We propose two novel strategies to improve the scheduling by exploring the bandwidth utilization from the perspectives of bandwidth occupation amount and time. Besides, a learning-based method which utilizes both the flow path delay and the network function-related delay to improve the VNF placement is proposed.

3 Network Model and Problem Formulation

Different applications require different services in which the flow needs to traverse the network functions in sequence in a specified service chain. For example, in data-centre network, a flow may need first go through an intrusion detection system (IDS) for anti-virus checking, and then go through a firewall to block the flooding. The IDS and firewall should be placed to the appropriate locations in the network, and the flow path should be well scheduled to go through the placement locations of IDS and firewall, so that the network is able to obtain the minimized delay.

The joint problem is modeled to find the optimal paths for the data flows and the placement locations for the VNFs in the network .

3.1 Network Model

The network is denoted as a graph $G(V, E)$, where V represents the set of network nodes, and E represents the set of links. The link between node $i \in V$ and $j \in V$ is denoted as $(i, j) \in E$. The available resource of node i to place VNFs is denoted as R_i. The delay and bandwidth capacity between two connecting nodes i and j are denoted by $l_{i,j}$ and $C_{i,j}$.

For a flow f in the flow set $\mathcal{F}$, the bandwidth requirement of f is denoted as b_f, the related service chain policy is denoted as p_f. Let $n_{f,k}$ denote the kth network function in p_f, $k = 1, ..., N_f$, where N_f is the number of network functions in p_f. The ingress node and egress node of a flow f in the network is denoted as f_{in} and f_{eg} respectively.

For a given network function $n_{f,k}$ in p_f, if it is the network function of the tth type, all the instances of the type of network function are able to implement the functionality of $n_{f,k}$ in p_f. The sth instance of the t type of network function is denoted as $i_{t,s}$, $s = 1, ..., N_t$, $t = 1, ..., T$, where N_t is the number of instances of the tth type of network function, T is the number of types of network functions. Suppose the maximum value of N_t is N_{ins}. The resource required to place $i_{t,s}$ is represented by $r_{t,s}$. We introduce several binary variables of this model. The binary variable $x_{i,j,f}$ is used to indicate whether f goes through the link (i, j). The value of $x_{i,j,f}$ is 1 when f goes through the link, and 0 otherwise. The binary variable $y_{f,k,i}$ is used to indicate whether $n_{f,k}$ is placed to node i. The value of $y_{f,k,i}$ is 1 when $n_{f,k}$ is placed to node i, and 0 otherwise. The binary variant $z_{t,s,i}$ is used to indicate whether the sth instance of the tth type of network function is placed to node i. The value of $z_{t,s,i}$ is 1 when $i_{t,s}$ is placed to node i, and 0 otherwise. The binary indicator $w_{f,k,t}$ is used to represent whether the kth network function in p_f is the tth type of

network function. The value of $w_{f,k,t}$ is 1 when $n_{f,k}$ is the tth type of network function, and 0 otherwise.

The denotations of this paper are summarized in Table 1.

3.2 Problem Formulation

The flow needs to occupy a certain amount of bandwidth when traversing in the network. As the network bandwidth is limited, the flow path should be carefully scheduled, and the placement locations of the network functions should be carefully determined so that the predefined network performance can be achieved. In this paper, we formulate the flow path scheduling and VNF placement as a joint optimization to minimize the network delay. Given a flow set $\mathcal{F}$ and a service chain set $\mathcal{S}$, the objective is to decide the placement location (node) of every network function in the service chain and find the path for each flow $f \in \mathcal{F}$ so that the network is able to achieve the minimized delay.

Table 1 Denotations of the variants

Denotation	Definition
$G(V, E)$	the network graph
(i, j)	the link between two connected node, $(i, j) \in E$
$C_{i,j}$	the bandwidth capacity of the link between node i to node j
$l_{i,j}$	the delay between node i to node j
R_i	the resource of node i to place network functions
f	flow
$\mathcal{F}$	flow set
b_f	bandwidth requirement of f, $f \in \mathcal{F}$
p_f	service chain specified by flow f, $f \in \mathcal{F}$
$n_{f,k}$	the kth network function in p_f
N_f	the number of network functions in p_f
n_t	the tth type of network function, $t = 1, ..., T$
$i_{t,s}$	the sth instance of the tth type of network function, $s = 1, ..., N_t$, $t = 1, ..., T$
N_t	the number of instances of the tth type of network function, $t = 1, ..., T$
T	the number of types of network functions
$r_{t,s}$	the resource required of placing $i_{t,s}$
f_{in}	the ingress node of f, $f \in \mathcal{F}$
f_{eg}	the egress node of f, $f \in \mathcal{F}$
D	the total end-to-end delay of all flows
$w_{f,k,t}$	binary indicator whose value is 1 when $n_{f,k}$ is the tth type of network function, and 0 otherwise
$x_{i,j,f}$	binary variable whose value is 1 when f goes through (i, j), and 0 otherwise
$y_{f,k,i}$	binary variable whose value is 1 when $n_{f,k}$ is placed to i, and 0 otherwise
$z_{t,s,i}$	binary variable whose value is 1 when $i_{t,s}$ is placed to i, and 0 otherwise

The flow in the network firstly needs to traverse to the ingress node f_{in}, then the network nodes placed with the network functions in the service chain in sequence, and lastly the egress node f_{eg}. The accumulated delay along the traversing nodes is the flow path delay, which is denoted as d_f and can be calculated by

$$\begin{aligned} d_f = & \, l_{f_{in},i^1} * w_{f,1,t'} * \left(\sum_{s'} z_{t',s',i^1} \right) \\ & + l_{i^{N_f},f_{eg}} * w_{f,N_f,t''} * \left(\sum_{s''} z_{t'',s'',i^{N_f}} \right) \\ & + \sum_{k=1}^{N_f-1} l_{i^k,i^{k+1}} * w_{f,k,t^*} * \left(\sum_{s^*} z_{t^*,s^*,i^k} \right) \\ & * w_{f,k+1,t^\dagger} * \left(\sum_{s^\dagger} z_{t^\dagger,s^\dagger,i^{k+1}} \right), \end{aligned} \tag{1}$$

where t', t'', t^* and $t^\dagger$ represent the types of network functions, s', s'', s^* and $s^\dagger$ are the corresponding instances of the type. The first item in Eq. 1 is called *ingress delay*, which means the delay from ingress node f_{in} to the node (i^1) placed with the first network function in p_f. The second item is called *egress delay*, which means the delay from the node (i^{N_f}) placed with the last network function in p_f to egress node f_{eg}. The third item is called *middle delay*, which means the delay from the node placed with the first network function to the node placed with the last network function in p_f. The flow path delay varies to the changes of the delay between two connected nodes, the service chain specifications and the placement locations of the network functions. The total delay D is obtained by summing the path delay of all the flows, which is represented as

$$D = \sum_f d_f. \tag{2}$$

The objective is to obtain the minimized D by efficient VNF placement and flow path scheduling within the various constraints of the network.

- ***Link Bandwidth Constraint***. The accumulated bandwidth requirements of the flows going through a link should not exceed the bandwidth capacity of the link, which is represented as

$$\sum_f b_f * x_{i,j,f} \leq C_{i,j}, \forall (i, j) \in E. \tag{3}$$

- ***Node Resource Constraint***. The resource required by the network functions that placed to a node should not

exceed the available resource of the node, which is represented as

$$\sum_{t=1}^{T}\sum_{s=1}^{N_t} r_{t,s} * z_{t,s,i} \leq R_i, \forall i \in V. \tag{4}$$

– ***Function Instance Constraint***. To reduce the placement cost, the number of instances of a type of network function that placed to a node should not exceed 1, which is represented as

$$\sum_{s=1}^{N_t} z_{t,s,i} \leq 1, \forall i \in V, \forall t \in T. \tag{5}$$

– ***Service Chain Constraint***. The flow f needs to traverse the nodes placed with the network functions in the service chain in sequence. If the kth network function in the service chain is the tth type of network function in the network function set, there should be a node placed with the network function, which is represented as

$$\sum_{k} w_{f,k,t} * \left(\sum_{s} z_{t,s,i}\right) = N_f, \forall f \in \mathcal{F}. \tag{6}$$

– ***Function Type Constraint***. For an instance of any type of network functions, there should be a placement location for it, which is represented as

$$\sum_{s=1}^{N_t}\sum_{i \in V} z_{t,s,i} = N_t, \forall t \in T. \tag{7}$$

The problem is formulated to minimize the network delay when considering all the constraints, which is represented as

$$\begin{aligned} &\min \quad D, \\ &s.t.\ (3), (4),\ (5), (6), (7). \end{aligned} \tag{8}$$

The VNF placement problem involved with the service chain requirement is proved to be NP-hard in many works [6, 18]. Combined with the flow path scheduling, it is hard to find the optimal solution in polynomial time.

4 Design of JOSP

The network delay is a main factor that affects the user's quality of experience. For example, 40% viewers in YouTube will quit watching the videos if there are rebufferings caused by the delay [29]. The search volume of Google will decrease by 0.74% if the response time increases by 400ms [30]. The network delay is highly related to the locations of VNFs and the path scheduling schemes in the requirement of the service chain. We present Joint Optimization of flow path Scheduling and virtual network function Placement (JOSP), an innovative method to figure out the scheduling and placement problem presented in Eq. 8 to achi-eve the approximate optimal network delay.

Two innovative strategies, i.e., DES and LSFD, are used to schedule the flow paths according to the service chain requirements and the placement locations of the VNFs. Specifically, DES aims to reduce the delay by adding the penalty to the link due to the link occupation status. The network delay is reduced, and the bandwidth utilization is improved. LSFD takes the flow duration of flows into account in the path scheduling and makes the bandwidth utilization to be more reasonable by avoiding the longtime occupation of the congested link. Then, JOSP calculates the flow path delay and network function-related delay. A reinforcement learning method is presented to decide the placement locations of VNFs in JOSP, where both of the flow path delay and network function-related delay are used to update the reward and persistently improve the placement performance by adjusting the placement locations of VNFs. The overview of JOSP is shown in Fig. 1.

4.1 DES

The link bandwidth occupation in the path scheduling for current flow will further influence the path selection for other flows and even bring extra congestion and increase the total delay. For example, as shown in Fig. 2, the highlighted coloured curves represent the flow paths (there are five flows highlighted). The bandwidth requirement and the composing links of the flow path are shown in Table 2. The link occupation is further shown in Table 3. As can be found, the bandwidth occupation of the links are different as the links are used by different flows. For the occupied link, if the link bandwidth is exhausted, it cannot be used in the path scheduling for the subsequent flows. The bandwidth occupation should be well scheduled to avoid congestion and reduce the delay.

With more hosting flows and more consumed bandwidth of a link, the processing resources of the related nodes (servers) and the remaining bandwidth of the link will be less for the subsequent flows to use [9, 31]. If the link turns to a bottleneck link, it will degrade the performance of flow path scheduling [22], there will be more chances to cause congestion if the subsequent flows are routed to it. Therefore, the unreasonable occupation of the link bandwidth should be avoided.

In order to balance the utilization of link bandwidth, we present the idea of delay scaling, which adds the penalty to the link delay according to a scaling factor of the link. The scaling factor of a link is measured by the bandwidth occupation situation of the link. The link with

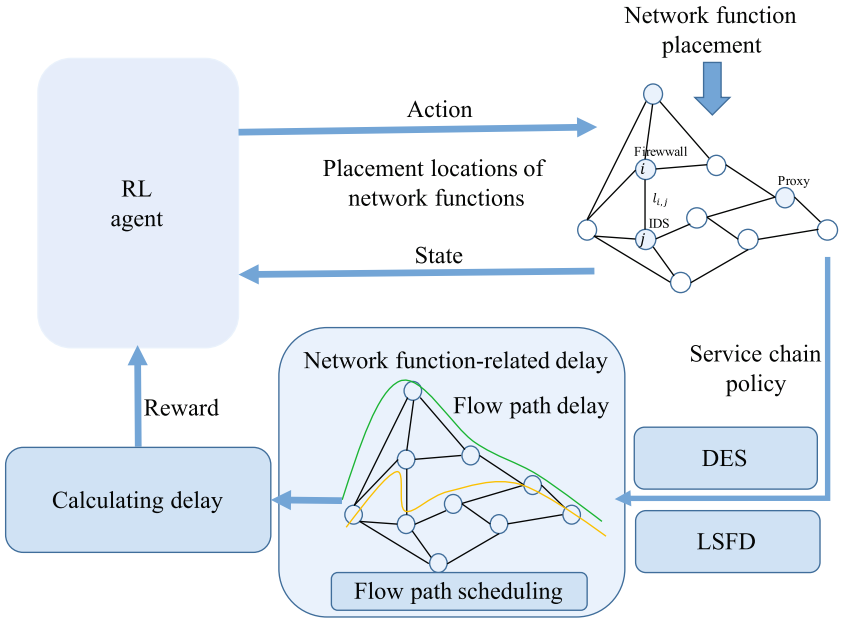


Fig. 1 Overview of JOSP

larger consumed bandwidth and less remaining bandwidth will be allocated with a larger factor. Because there will be less capacity for other flows to use the link in path scheduling, which increases the risk of congestion.

The scaling factor of a link (i, j), denoted by $s_{i,j}$, can be measured by the ratio of consumed bandwidth to the remaining bandwidth of the link,

$$s_{i,j} = \frac{\sum_f x_{i,j,f} * b_f}{C_{i,j} - \sum_f x_{i,j,f} * b_f}. \tag{9}$$

If the bandwidth of a link is exhausted, it should never be used in the path scheduling for the subsequent flows as it is meaningless to calculate the scaling factor.

The delay of a link is scaled by multiplying the scaling factor to the initial link delay. Let $l'_{i,j}$ denote the scaled delay of link (i, j), $l'_{i,j} = s_{i,j} * l_{i,j}$, where $l_{i,j}$ is the initial delay of the link,. For the incoming flow, the accumulated scaled delay of all links along the chosen path is calculated. In other words, the delay scaling increases the cost of using the link bandwidth. The total cost of using the link bandwidth along the path is obtained. The flow path is scheduled under the condition that the total cost is minimized. Therefore, the bandwidth is able to be used in a more reasonable way instead of simply choosing the shortest available path. The DES is able to even out the congestion condition of involved links in the network. The pseudo-code of DES is given in Algorithm 1.

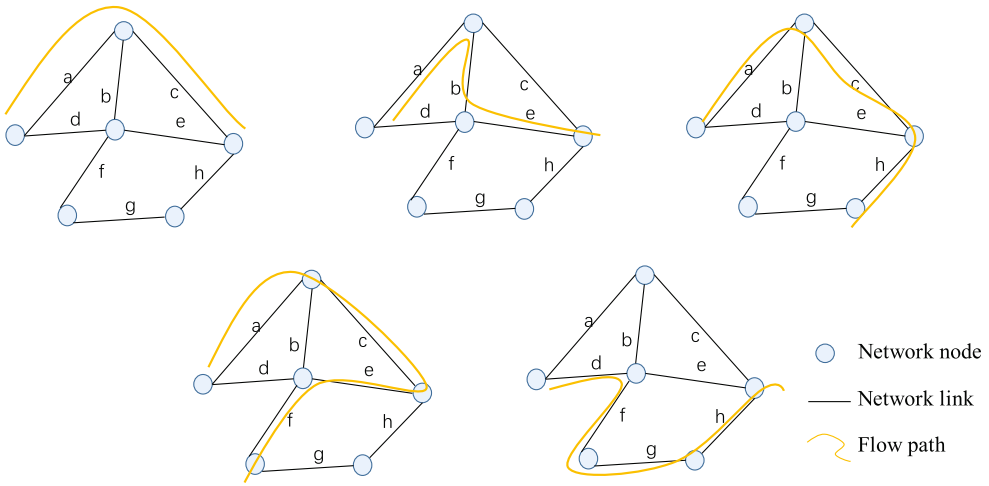


Fig. 2 Illustration of flow path and the occupation of links, the colored curve denotes the flow path composed of the connected links

Table 2 Bandwidth requirement and the composing links of flow path

Flow	Included Link	Bandwidth occupation
f_1	a, c	b_{f1}
f_2	a, b, e	b_{f2}
f_3	a, c, h	b_{f3}
f_4	a, c, e, f	b_{f4}
f_5	d, f, g, h	b_{f5}

Algorithm 1 DElay scaling.

Input: Network topology $G(V, E)$, link bandwidth capacity $C_{i,j}$, delay between two connected nodes $l_{i,j}$, existing flow set $\mathcal{F}'$

Output: Scaled delay $l'_{i,j}$

```
//Scale links' delay according to link utilization
for (i, j) ∈ E do
    Calculate bandwidth occupation of the link
    if link bandwidth is exhausted then
        continue
    else
        s_{i,j} = (Σ_{f∈F'} x_{i,j,f} * b_f) / (C_{i,j} − Σ_{f∈F'} x_{i,j,f} * b_f)
    end if
    l'_{i,j} = s_{i,j} * l_{i,j}
end for
return l'_{i,j}
```

4.2 LSFD

The duration of flows in the network varies with the service type. For example, streaming video usually lasts long [32] while anomaly detection has a short duration [33]. The flow duration often follows a heavy-tail distribution [34]. There are many efforts of research aiming at tracking the flow duration by various methods, e.g., real-time monitoring, estimation and detection theory [35, 36]. Flows could be differentiated by the flow duration to fulfil different purposes, for example, security checking, network management, etc. [37–39].

A flow with a long duration will occupy the link bandwidth for a longer time. In particular, the longtime occupation of the bandwidth of bottleneck link will cause congestion. Because if the subsequent flow needs to use the link bandwidth, it has to wait until the flow with a long duration finishes service and releases the bandwidth. However, the short flow only occupies the bandwidth in a shorter time, which does not impede the utilization of the related bandwidth for other flows. Therefore, the long flow and short flow should be applied with different bandwidth occupation strategies.

4.2.1 Long Flows

If the duration of a flow t_f exceeds a predefined threshold η, the flow is identified as a long flow, i.e., the flow f is seen as a long flow if

$$t_f \geq \eta. \tag{10}$$

If a link is demanded by many other flows or the bandwidth of the link is heavily utilized compared to its bandwidth capacity, it has the possibility to cause congestion.

LSFD schedules the path of long flow to the links with enough remaining bandwidth and would not be required by many other flows, called *Cool Track*. Specifically, the *Cool Track* should satisfy the following two conditions.

– The remaining bandwidth of the link should not be small. The available bandwidth of a link is obtained by subtracting the demanded bandwidth of the flows using the link from the bandwidth capacity of the link. The ratio of the remaining bandwidth to the bandwidth capacity of a link could be used to indicate the bandwidth occupation condition of the link. Choosing the link with a larger ratio to be part of the path of the long flow would not influence the path scheduling of other flows, as there are still available bandwidth if other flows want to use the link. LSFD schedules the path of the long flow to the link that the ratio of the remaining bandwidth to the bandwidth capacity of

Table 3 The link occupation

Link	Number of Flows	Flows Running on It	Bandwidth Capacity	Occupied Bandwidth
a	4	f_1, f_2, f_3, f_4	C_a	$b_{f1} + b_{f2} + b_{f3} + b_{f4}$
b	1	f_2	C_b	b_{f2}
c	3	f_1, f_3, f_4	C_c	$b_{f1} + b_{f3} + b_{f4}$
d	1	f_5	C_d	b_{f5}
e	2	f_2, f_4	C_e	$b_{f2} + b_{f4}$
f	2	f_4, f_5	C_f	$b_{f4} + b_{f5}$
g	1	f_5	C_g	b_{f5}
h	2	f_3, f_5	C_h	$b_{f3} + b_{f5}$

the link is bigger than a threshold γ_1. The decision condition of whether choosing a link as part of the path for a long flow when considering the influence of bandwidth occupation on the link is represented as, if

$$\frac{C_{i,j} - \sum_{f \in \mathcal{F}'} x_{i,j,f} * b_f}{C_{i,j}} \leq \gamma_1, \tag{11}$$

link (i, j) is excluded from the links that could compose the path for long flow, where $\mathcal{F}'$ is the set of flows that have already been scheduled with paths.

– The number of flows on the link should not be large. The number of flows running on a link indicates the processing resource required for the related nodes (servers). If a link has more flows running on it, the required processing resources will be more. The path for long flow should avoid using the link that is processing with too many other flows because the long flow will occupy the processing resources for a longer time. The ratio of the number of flows on the link to the number of all flows is used to indicate the busyness of the link. If the ratio is smaller than a threshold γ_2, it means there are not so many flows employing the link as part of their paths. However, if the ratio is bigger than the threshold, it means the link is busy processing with other flows. LSFD schedules the path of long flow to the links with a small ratio, which would not induce disturbance to the path scheduling for other flows. The decision condition of whether choosing a link as part of the path for a long flow when considering the influence of the number of flows on the link is represented as, if

$$\frac{\sum_{f \in \mathcal{F}'} x_{i,j,f}}{F_{num}} \geq \gamma_2, \tag{12}$$

link (i, j) is excluded from the links that could compose the path for the long flow.

The unreasonable longtime occupation of the link bandwidth and processing resource are avoided when scheduling the path of long flow to the *Cool Track*. The *Cool Track* is obtained by choosing the shortest path with the available link bandwidth except the links that satisfy Eqs. 11 and 12.

4.2.2 Short Flows

If the duration of a flow t_f is less than a predefined threshold η, the flow is identified as a short flow, i.e., the flow f is seen as a short flow if

$$t_f < \eta. \tag{13}$$

LSFD schedules the short flow to the path with the shortest delay, which is called *Fast Track*. The network is able to hold more amount of flows in the same period of time to reduce the delay. The short flow is able to achieve low delay and would not occupy the link bandwidth for a long time. The occupied link bandwidth by short flow will be released once the short flow finishes the service. The released bandwidth is able to be used by other flows, which in turn helps the other flows to reduce the delay. With more short flows being scheduled to *Fast Track*, the delay is able to be reduced to a larger extent. The *Fast Track* is obtained by choosing the available path with minimum delay by Dijkstra algorithm [40].

The path scheduling algorithm based on DES and LSFD is shown in Algorithm 2.

Algorithm 2 Path scheduling.

Input: Network topology $G(V, E)$, link bandwidth capacity $C_{i,j}$, existing flow set $\mathcal{F}'$, incoming flow set $\mathcal{F}$
Output: Flow path set $paths$

```
//Assign paths for flows according to delay scaling and long-short flow differentiating
for an incoming flow f ∈ F do
    if t_f < η then
        Assign the path with shortest delay to f
        Record i^1, ..., i^k, ..., i^{N_f}
    else
        for link (i, j) ∈ E do
            if (C_{i,j} − Σ_{f∈F'} x_{i,j,f} * b_f) / C_{i,j} ≤ γ1 then
                Remove link (i, j)
            end if
            if (Σ_{f∈F'} x_{i,j,f}) / F_num ≥ γ2 then
                Remove link (i, j)
            end if
        end for
        Assign the shortest path to f using available links
        Move f from F to F'
        paths = paths ∪ {f_in, i^1, ..., i^k, ..., i^{N_f}, f_eg}
    end if
end for
return paths
```

4.3 VNF Placement

Improper placement of VNFs would bring trouble to the flow path scheduling because once the placement is decided, flows need to route paths according to the locations of VNFs specified by the service chains. As a result, more resources will be required to process the overwhelmed traffic and extra congestion will be introduced. Both of which will dramatically decrease the utilization as well as the per-

formance of the network. Therefore, the VNF placement scheme needs to be carefully considered to facilitate the flow path scheduling so that the service chain functionalities and delay requirements from flows are satisfied.

Based on the observation that the reinforcement learning method is very efficient in solving the decision making problems [34], JOSP presents the improved Q-learning [41] (a reinforcement learning method) based VNF placement scheme to find the optimal placement locations of VNFs to minimize the delay. The framework is shown in Fig. 3.

The agent collects state information (the previous placement location) from the environment and make decisions on the next placement locations of the VNFs. The flow path is then obtained according to the placement locations and service chain requirement. The flow path delay and the network function-related delay are feedback to calculate the reward to the agent to help improve the placement.

The method has three elements, i.e., a set of states, a set of actions and the reward. The state is all the possible placement locations of the VNF. The action is to decide the next placement location, the action also brings the reward and new state. The reward is calculated to measure the action. Let s, a, r to denote the *state*, *action* and *reward* at the current step respectively. The action is made by maximizing the accumulated discounted reward with a discount factor γ, which determines the importance of the future reward. After the action, the state turns to a new state s'. The expected reward is denoted by $Q(s, a)$. The updating of $Q(s, a)$ is represented as [41]

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a)+\alpha\left[r+\gamma\max_{a'}Q(s',a')\right], \quad (14)$$

where α is the learning rate, it represents the ratio of the newly learned information to the acquired $Q(s, a)$. The value of α is between $[0, 1]$. If the value is 0, it means $Q(s, a)$ is totally obtained from the old information. If the value is 1, it means everything is newly learned.

For every VNF, there is a matrix to store the values of $Q(s, a)$. For every possible action, the value in the matrix represents the recommended value of placing the VNF to the corresponding location at the current state. However, the traditional Q-learning based method update the Q-value by making the action based on the optimal Q-value. For the action that has been made before, the value may be over estimated. In turn, for the action that has not been made, the value will stay low. When the action space is very large, it will be quite difficult to make the optimal action. We present the improved Q-learning based method to the VNF placement in JOSP, shown in Algorithm 3. The *state*, *action* and *reward* are specified as follows.

Algorithm 3 VNF placement.

Input: Network topology $G(V, E)$, the placement capacity $R_i, i \in V$, the resource required of placing the sth instance of the t type of network function $r_{t,s}$

Output: The placement location of network function $n_{t,s}$ The Q-table for inference

```
for the tth type of VNF do
    for the sth instance do
        Initialize Q-table=O, state s = {random(i)}
        for each step in the Training Cycle do
            Choose action from state s
            if random(0,1)> ϵ then
                Decide the placement location according to
                the greedy manner
            else
                Sort the placement locations by Q-values
                for the placement location do
                    Check R_i of the placement location
                    if R_i > r_{t,s} then
                        Place the VNF to the location
                        R_i ← R_i − r_{t,s}
                    else
                        Continue
                    end if
                    Break
                end for
            end if
            Observe new state s'
            Calculate reward according to Eq. 15
            Calculate the value in Q-table
            Update Q-table
            State transition s ← s'
        end for
        return Q-table
    end for
end for
```

– ***State***. Every possible placement scheme in the network is seen as a state. The number of placement locations for the VNF is the number of states. There is a state transition when VNF is placed to a new location. If the VNF is placed to a location, the required resource is subtracted from the placement location. If the VNF is removed from a location, the required resource will be released. When the resource capacity of a placement location is exhausted, it should not be placed with any other VNFs.

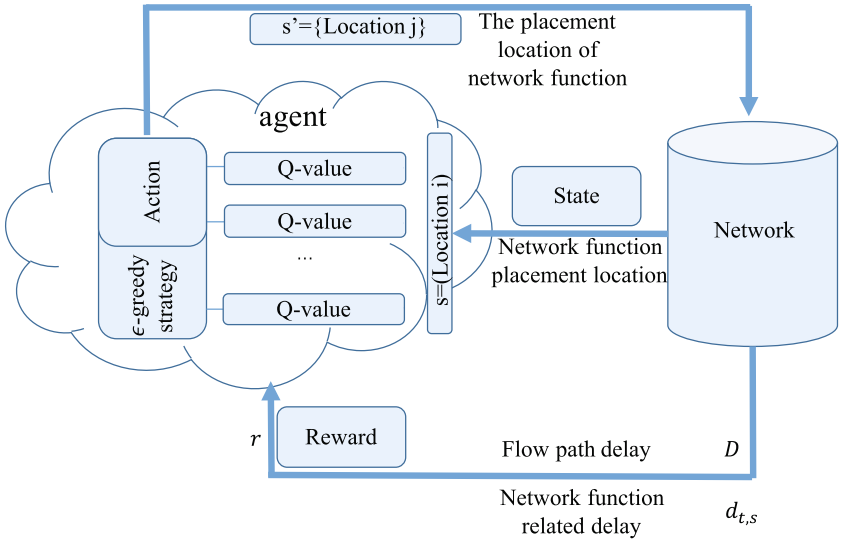


Fig. 3 Overview of the VNF placement method

- ***Action***. The action is to place the VNF to the location by the ϵ-greedy strategy [41] based on the current state. The number of placement locations for the VNF is also the number of actions. When the VNF is placed to a location, it turns to the new state, which is called the state transition. To avoid the over estimate of the Q-value, JOSP uses a parameter ϵ to control the selection of the placement location of the VNF. In each action step, a random number is generated to compare with the parameter ϵ. If the random number is larger than ϵ, the placement location of the VNF will be decided by a greedy manner. The method makes the VNFs to be placed to the locations that generate the shortest delay for the relevant flows. Otherwise, the placement location of the VNF will be decided by choosing the action with the largest Q-value in the Q-table.
- ***Reward***. The network delay is taken as the metric to optimize the VNF placement. The placement location that results in poor performance (high delay) has less probability of being considered in the future, while the placement location that reduces the delay will be encouraged. The reward is calculated after the action. JOSP takes the total end-to-end delay as well as the network function-related delay to calculate the reward. The total end-to-end delay represents the whole effect of VNF placement. The network function-related delay indicates whether the placement location of a single VNF is reasonable. The total end-to-end delay obtained under the current action is denoted by D^{cur}, and the value obtained under the last action is denoted by D^{pre}. The network function-related delay under current action is denoted by $d_{t,s}^{cur}$, and the value obtained under the last action is denoted by $d_{t,s}^{pre}$. The two different kinds of delay are combined together to calculate the reward to the agent to make the placement decision in the next step. The reward is denoted by

$$r = \sigma_1 \left(D^{pre} - D^{cur}\right) + \sigma_2 \left(d_{t,s}^{pre} - d_{t,s}^{cur}\right), \tag{15}$$

where σ_1 and σ_2 are the coefficients.

After the training, the obtained Q-table is used for the inference of the placement of the network function.

4.4 Delay Calculation

The total end-to-end delay can be obtained by summing the path delay of all the flows, which is represented as

$$D = \sum_{f \in \mathcal{F}} d_f, \tag{16}$$

where d_f is the path delay of flow f. d_f is obtained by adding the delay between the connected nodes along the flow path.

Besides the flow path delay, the network function-related delay is also considered to calculate the reward. The network function-related delay is defined as the minimum delay from the node placed with the VNF to the nodes placed with its adjacent neighbouring VNFs in the service chains of all flows. For example, as shown in Fig. 4, assume there are two service chains, {*IDS, Firewall*} and {*Firewall, Proxy*}. The network function-related delay of *Firewall* is obtained by calculating the summed minimum delay from the placement node of *Firewall* to the placement nodes of *IDS* and *Proxy*, shown as the green curve in Fig. 4. The network function-related delay is able to help to evaluate the influence of its placement location to reduce the network delay.

If the service chain specified by flow f contains a network function of the tth type (n_t) in the network function

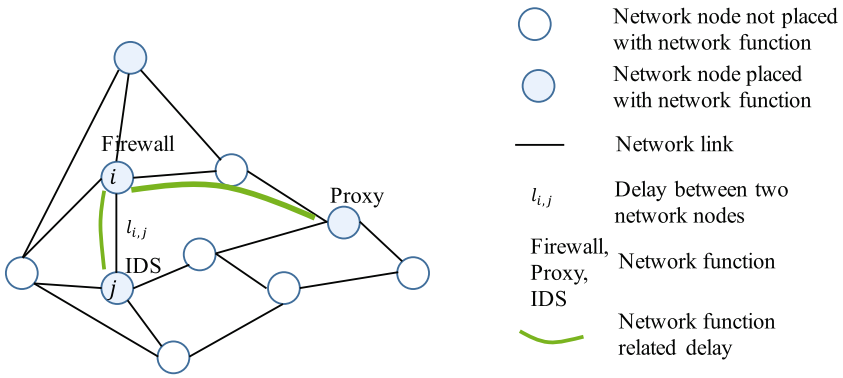


Fig. 4 Illustration of the network function-related delay

set, all of the instances of that type of network function placed in the network are able to fulfill the service requirement of f for the network function. Namely, if n_t is in the service chain of flow f, e.g., it is the kth network function in the service chain $(n_{f,k}, k = 1, .., ., N_f)$, then all the instances of n_t are qualified to act as $n_{f,k}$. If n_t is the network function of $n_{f,k}$ and $i_{t,s}$ is placed to node i, the $i_{t,s}$ related delay $d_{t,s}$ can be represented as

$$d_{t,s} = \sum_f min_u \left\{ l^f_{u,i} * y_{f,k-1,u} * w_{f,k,t} * z_{t,s,i} \right\} + \sum_f min_j \left\{ l^f_{i,j} * w_{f,k,t} * z_{t,s,i} * y_{f,k+1,j} \right\}, \quad (17)$$

where node u is placed with the network function $n_{f,k-1}$, node j is placed with the network function $n_{f,k+1}$.

The Eq. 17 ensures the following two conditions: i) $n_{f,k-1}$ is the predecessor network function of the concerned network function $i_{t,s}$ in the service chain of f, $n_{f,k+1}$ is the successor network function of the concerned network function $i_{t,s}$ in the service chain of f. ii) If there are multiple nodes placed with the same type of network function of $n_{f,k-1}$ or $n_{f,k+1}$ ($n_{f,k-1}$ or $n_{f,k+1}$ has multiple instances placed on different nodes), the minimum delay between the nodes placed with $n_{f,k-1}$ or $n_{f,k+1}$ and node i is added to the $i_{t,s}$ related delay. If $i_{t,s}$ is the first network function in the service chain, $n_{f,k-1}$ means the ingress node of f. If $i_{t,s}$ is the last network function in the service chain, $n_{f,k+1}$ means the egress node of f.

The pseudo-code to calculate the network function-related delay is shown in Algorithm 4.

4.5 Complexity Analysis

The DES needs to scale the delay of all the links in the network to obtain the logic topology. Therefore, the time complexity is $\mathcal{O}(|\mathcal{F}||E|)$, where $|\mathcal{F}|$ is the number of flows, $|E|$ is the number of links in the network.

The LSFD takes different scheduling strategies according to the flow duration, and all the links should be conducted with the link utilization check. The time complexity of finding the shortest path for a flow is $\mathcal{O}(|V| \log |V| + |E|)$ [40, 42], where $|V|$ is the number of nodes in the network. Therefore, the total time complexity of the flow path scheduling method is $\mathcal{O}(|\mathcal{F}|(|V| \log |V| + 2|E|))$.

Algorithm 4 Network function-related delay.

Input: Network topology $G(V, E)$, delay between two connected nodes $l_{i,j}$, existing flow set $\mathcal{F}'$
Output: network function-related delay $d_{t,s}$

```
//Calculate the network function-related delay
for each instance of each type of network function do
    d_{t,s} = 0
    for f ∈ F' do
        if the type is in the service chain of f then
            for each successor instance do
                d_{t,s} = Σ_{f∈F'} min_u{l^f_{u,i} * y_{f,k-1,u} * w_{f,k,t} * z_{t,s,i}}
            end for
            for each predecessor instance do
                d_{t,s} = d_{t,s} + Σ_{f∈F'} min_j{l^f_{i,j} * w_{f,k,t} * z_{t,s,i} * y_{f,k+1,j}}
            end for
        end if
    end for
end for
return d_{t,s}
```

The VNF placement uses the Q-learning to decide the placement location of each instance of network function, the time complexity is $\mathcal{O}(|V|^2)$ [41]. The number of all the instances of network functions is $\mathcal{O}(T N_{ins})$, where N_{ins} is the maximum number of instance for a network function in the function set. The time complexity of calculating the network function-related delay is $\mathcal{O}(2N_{ins}|\mathcal{F}|)$.

5 Evaluations

To evaluate the effectiveness of the proposed method to reduce the network delay, we perform various experiments in different network environment.

5.1 Evaluation Setup

We evaluate the proposed method under three different network topologies from the Stanford Network Analysis Project (SNAP) [43]. Topology A represents the campus network, which has a medium scale of the connected nodes and links. Topology B and Topology C represent classical fat-tree data-centre networks with different scales. The details of the three topologies are shown in Table 4.

The evaluation is implemented in the Intel i7 work station with 2.6 GHz, 6-core CPU and 16GB 2400 MHz DDR4 memory. Fast Network Simulation Setup is used to setup the network environment and the flow workload. The bandwidth demand of the flow is uniformly distributed between (0,50] Mbps. The bandwidth capacity of the link is assumed to be 100 Mbps. The Q-learning model for network function placement is trained by 600 rounds before it becomes optimal and can be used for inference. The delay between two connected nodes (including the propagation delay and the processing delay) is assumed to be randomly distributed with a maximum value of 20ms. The flow duration in the campus and data-centre networks is assumed to follow the Pareto distribution [32], which is a type of heavy-tail distribution. The key parameters are described in Table 5.

A service chain set composed of 50 different service chains is used in the evaluation. The service chain set has nine types of VNF, i.e., the load balancer, intrusion detection system, intrusion prevention system, proxy, monitoring, firewall, WAN optimizer, network address translation and domain name system. The number of VNFs in a service chain is variable in the evaluation. The VNF in the service chain is randomly selected from the network function set. The number of instances of a type of network function and the placement capacity of a node are set to be 3.

5.2 Benchmarks

We compare JOSP with the following methods to validate its effectiveness.

Table 4 Evaluation topologies

Topology	Number of nodes	Number of links
A	49	118
B	101	226
C	203	612

Table 5 Values of parameters

Parameter	Value	Description
$l_{i,j}$	20ms	the maximum delay between two connected nodes
$C_{i,j}$	100Mbps	the maximum link bandwidth capacity
b_f	50Mbps	the maximum bandwidth requirement from flows
α	0.01	the learning rate
γ	0.9	the discount factor
σ_1	0.0001	the coefficient of the total delay to calculate the reward
σ_2	0.001	the coefficient of the network function-related delay to calculate the reward
ϵ	0.7	the greedy control factor

- ***BS***: BS [16] is a state-of-the-art method that adopts reinforcement learning to solve the VNF placement to minimize the delay. BS uses the flow path delay to calculate the reward to adjust the placement locations of VNFs. However, the network function-related delay is not involved. It schedules the paths of flows by the shortest path first strategy.
- ***DES***: JOSP is also compared to the method that solely adopts the DES strategy to reveal its effectiveness. In DES, the delay between the connected nodes is scaled by multiplying a scaling factor to the initial delay according to the bandwidth occupation condition of the link. The DES strategy imposes the penalty of occupation of the link bandwidth that has the possibility to cause congestion in the flow path scheduling to reduce the delay.
- ***LSFD***: JOSP is also compared to the method that solely adopts the LSFD strategy. The LSFD dedicates to explore the heavy-tail distribution characteristics of the flow duration in flow path scheduling. It schedules the path of long flow to *Cool Track* and schedules the path of short flow to *Fast Track*. It avoids the longtime occupation of the bandwidth that is demanded by many other flows to reduce the delay.

5.3 Performance Metrics

The proposed method JOSP is compared with the benchmarks by using the following metrics.

- ***Delay***: Delay has been a main factor to affect the user's quality of experience [44]. Both the total end-to-end delay and the single flow delay are used to evaluate the proposed method. The total end-to-end delay is obtained by adding the path delay of every single flow.

$$D = \sum_{f} d_f. \tag{18}$$

– ***Bandwidth Utilization***: This metric reveals the efficiency of the bandwidth utilization, which is represented as

$$U = \sum_{(i,j)\in E} \frac{\sum_f b_f * x_{i,j,f}}{C_{i,j}|E|}. \tag{19}$$

5.4 Performance Comparison

We first evaluate the average end-to-end delay achieved by JOSP and the benchmark methods in Fig. 5. The threshold for differentiating the long flow and short flow is set to be 30ms. γ_1 and γ_2 are both set to be 0.5. As can be seen from the figure, the proposed method JOSP achieves the lowest delay among all the methods. The delay achieved by JOSP is reduced by 40% on average in different topologies when compared with BS. JOSP obtains lower delay for two reasons. Firstly, JOSP measures the overall bandwidth occupation of the network and considers the influence of the occupation of the bandwidth by the current flow to the subsequent flows. It adds the penalty to the utilization of the link bandwidth that has the possibility to cause congestion. The bandwidth is used in a more reasonable way by minimizing the penalized cost. Secondly, JOSP considers the bandwidth occupation time and avoids the longtime occupation of the busy link that will induce the delay. The bandwidth used by a flow will not disturb the bandwidth utilization of other flows. In particular, the delay generated by waiting for the bandwidth to be released as it has already been occupied by long flows is avoided. The DES and LSFD consider one of the two factors mentioned above respectively. Therefore, the delay is smaller than BS but is larger than JOSP.

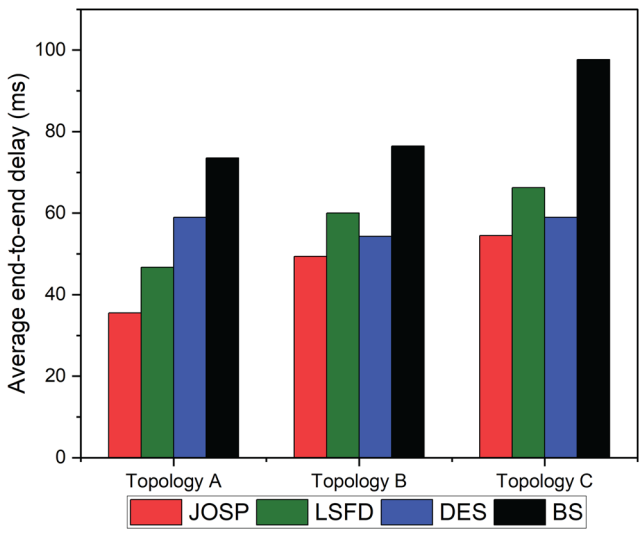


Fig. 5 Average delay achieved by different methods

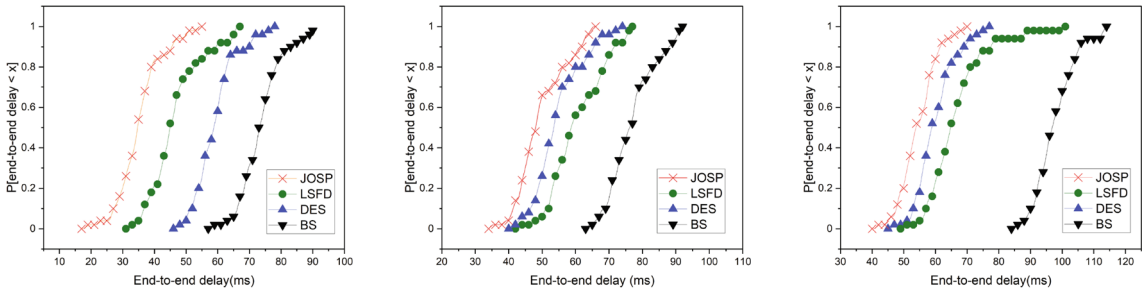
The DES uses a scaling factor to scale the delay of the link. The scaling factor is relevant to the bandwidth occupation situation. We find that when the scaling factor is set to be larger than 1.4, it cannot achieve the desirable delay. When the scaling factor is regulated to [1,0,1.4], the average delay is markedly decreased. This is because when the scaling factor is set to be a larger number, it will wipe out the influence of the actual delay between the connected nodes on finding the path with the shortest delay for the flow. The DES provides the metric of whether using a link according to the bandwidth occupation, which balances the bandwidth occupation of the network and reduces the congestion as well as the delay.

The LSFD schedules the paths of long flow and short flow separately. The long flow will not be scheduled to the link that the ratio of the remaining bandwidth to the bandwidth capacity is smaller than γ_1 or the ratio of the number of the flows using the link to the number of all flows is larger than γ_2. The short flow would be scheduled to path with the least delay, it will quickly release the bandwidth of the links on the flow path after finishing traversing and let the subsequent flows use the links to reduce the overall delay.

Moreover, JOSP uses the flow path delay as well as the network function-related delay to calculate the reward for the agent to adjust the placement locations of the VNFs. If the network function-related delay is very large, the path delay of the flows that require the service of the network function will also increase as they need to traverse to the placement location of the network function. By leveraging the network function-related delay to update the reward, JOSP makes the VNFs to be placed to the locations that generate less delay. BS only uses the flow path delay to calculate the reward when deciding the placement locations of the VNFs. It is not able to measure the delay generated by improper placement locations.

To show the performances of different methods in reducing the delay for every single flow in different network topologies, Fig. 6 plots the cumulative distribution function (CDF) curves. The x-axis shows the end-to-end delay, and the y-axis shows the ratio of flows whose delay is under a given value. As can be seen from the figure, the curves of JOSP are above the results of BS for the most delay values in different network topologies. It means the proposed method achieves smaller delay for most flows. It can also be found that JOSP performs better than the method that solely uses the DES or LSFD strategy.

Moreover, as can be seen from Fig. 6, LSFD performs better than DES in Topology A, while DES performs better than LSFD in the other two topologies. It indicates that the exploration of bandwidth occupation time is able to achieve less delay in the network with fewer nodes and edges. While



(a) Topology A (b) Topology B (c) Topology C

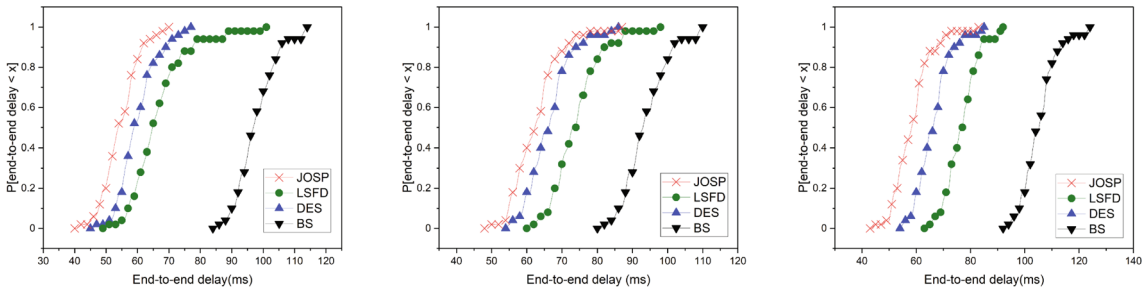
Fig. 6 Flow delay achieved by different methods in different topologies

in the network with more connections. The DES strategy is more efficient. This is because the network with more connections has high diversity in the flow paths, the DES strategy can find the optimal path by adding a penalty to the bandwidth occupation.

The threshold (η) to differentiate the long flow and short flow needs to be adjusted when the workload changes. We evaluate the performance of JOSP in Topology C when the workload changes. Topology C represents the large-scale data-centre network and has a huge uncertainty of the traffic, so the threshold η also needs to change. Figure 7 shows the performance of JOSP when η is 30ms, 50ms or 70ms with the varying workload. The threshold η is set to be 30ms when there are more flows with short duration. When there are more flows with a longer duration, the threshold η needs to be larger, say 70ms. As can be seen from the figure, the delay achieved by JOSP always outperforms the benchmark methods under different workloads by changing the threshold, which also validates the effectiveness of the proposed method.

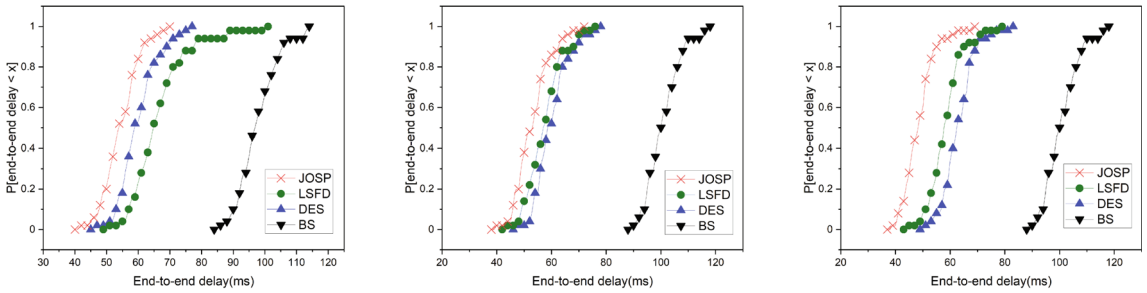
LSFD takes the link occupation situation into account when scheduling the path for the flow. The parameter γ_1 and γ_2 are used to judge the link occupation, and they have different impacts on the performance of reducing the delay. γ_1 is the threshold compared with the ratio of the remaining bandwidth to the bandwidth capacity. If the ratio is smaller than γ_1, the long flow cannot use the link under the LSFD strategy. γ_2 is the threshold compared with the ratio of the number of flows on the link to the number of all flows. If the ratio is larger than γ_2, the long flow cannot use the link under the LSFD strategy. We calculate the delay when applying different parameter values. The result is shown in Fig. 8. Figure 8(a) shows the delay when γ_1 and γ_2 are both set to be 0.5. When γ_1 is set to a smaller value, say $\gamma_1 = 0.3$, the performance gap between JOSP and LSFD becomes smaller, shown in Fig. 8(b). When the threshold is regulated to a smaller value, the long flow has more choices in path scheduling by LSFD, so the delay is lower. When γ_2 is set to a larger value, say $\gamma_2 = 0.7$, the performance gap between JOSP and LSFD becomes larger, shown in Fig. 8(c). When the threshold is regulated to a larger value, the LSFD strategy has less chance to be activated. Moreover, as can be found from the figure, the gap between JOSP and LSFD of Fig. 8(b) is smaller than that of Fig. 8(c), which means the bandwidth occupation is more competent to evaluate the influence of the LSFD strategy.

The bandwidth utilization is also an important indicator to evaluate the performance of the proposed method. The results are shown in Fig. 9. The bandwidth utilization is measured by calculating the occupied link bandwidth on the



(a) $\eta = 30$ms (b) $\eta = 50$ms (c) $\eta = 70$ms

Fig. 7 Flow delay achieved by different methods when the threshold for differentiating long flow and short flow changes



(a) $\gamma_1 = 0.5, \gamma_2 = 0.5$ (b) $\gamma_1 = 0.3, \gamma_2 = 0.5$ (c) $\gamma_1 = 0.5, \gamma_2 = 0.7$

Fig. 8 Flow delay achieved by different methods when the parameters to judge the link occupation change

flow path. To directly show the differences of the comparing methods, the results are divided by the bandwidth utilization of BS. As can be seen from the figure, the proposed method achieves higher bandwidth utilization than the benchmark method. This is because JOSP schedules the flows to more available links to avoid the congestion based on the link utilization situation and the bandwidth occupation requirement, thus reducing the delay caused by waiting for the link bandwidth to be released. In Topology A, the bandwidth utilization of LSFD is the highest. Topology A represents the campus network, which has fewer nodes and links. The flow path has fewer diversities, and the LSFD strategy has more chances to be used in the scheduling. Therefore, the bandwidth utilization gets higher in the path scheduling. The DES strategy improves the bandwidth utilization by adding a penalty to the link that has the possibility to generate congestion. The tradeoff is obtained by exploring the utilization of the network bandwidth to reduce the delay. With the DES and LSFD strategy, the proposed method can use the network bandwidth more efficiently.

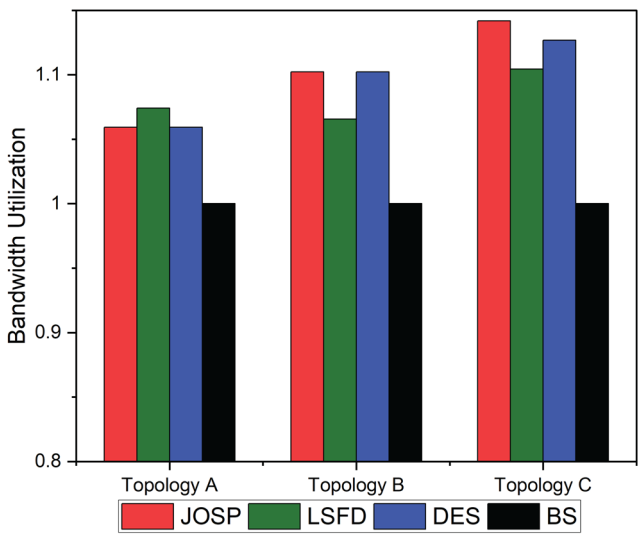


Fig. 9 Bandwidth utilization of different methods

6 Conclusion

In this paper, we have put forward a framework called JOSP to jointly optimize the flow path scheduling and VNF placement. We have presented two strategies, i.e., DES and LSFD, for scheduling and a learning-based method for the placement. DES adds the penalty to the occupation of the link bandwidth that has the possibility to generate congestion and LSFD differentiates the long flow and short flow in the path selection to explore the best utilization of the bandwidth to reduce the delay. Furthermore, empowered with the achieved flow path delay and network function-related delay to update the reward for the learning agent, JOSP adjusts the placement locations of the VNFs. The proposed JOSP has outperformed the existing methods by 40% on average in reducing the network delay.

In our future work, we plan to add the bandwidth occupation time of the flows as a feature to be learned without human intervention and use distributed learning-based methods to enhance the placement performance.

Acknowledgements This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 61872212, Grant 61902044, Grant 62072060 and Grant 62102053, the Australian Research Council Linkage Project under Grant LP190100676, the Natural Science Foundation Projects in Chongqing under Grant cstc2019jcyj-msxmX0442 and Grant cstc2019jcyj-msxmX0589, and the Chongqing Key Laboratory of Digital Cinema Art and Technology under Grant KJJ-CQ-2020007.

References

1. Do TX, Kim YH (2019) State management function placement for service-based 5G mobile core architecture. Mob Netw Appl 24(2):504–516
2. Sun G, Zhou R, Sun J, Yu HF, Vasilakos AV (2020) Energy-efficient provisioning for service function chains to support delay-sensitive applications in network function virtualization. IEEE Internet Things J 7(7):6116–6131
3. Deng Y, Zhang TH, Lou GN, Zheng X, Jin J, Han QL (2021) Deep learning-based autonomous driving systems: a survey of attacks and defenses. arXiv:2104.01789

4. Mo HM, Ding S, Yang SL, Zheng X, Vasilakos AV (2020) DLRRMS: Deep learning based respiratory rate monitoring system using mobile robots and edges. arXiv:2011.08482
5. Dibaei M, Zheng X, Xia Y, Xu X, Jolfaei A, Bashir AK, Tariq U, Yu D, Vasilakos AV (2020) Investigating the prospect of leveraging blockchain and machine learning to secure vehicular networks: A survey, Piscataway
6. Liu JQ, Li Y, Zhang Y, Su L, Jin DP (2017) Improve service chaining performance with optimized middlebox placement. IEEE Trans Serv Comput 10(4):560–573
7. Yang S, Li F, Trajanovski S, Chen X, Wang Y, Fu XM (2021) Delay-aware virtual network function placement and routing in edge clouds. IEEE Trans Mob Comput 20(2):445–459
8. Pei JN, Hong PL, Xue KP, Li DF, Wei D, Wu F (2020) Two-phase virtual network function selection and chaining algorithm based on deep learning in SDN/NFV-enabled networks. IEEE J Sel Areas Commun 38(6):1102–1117
9. Gvozdiev N, Vissicchio S, Karp B, Handley M (2018) On low-latency-capable topologies, and their impact on the design of intra-domain routing. In: Proc. of the 2018 conference of the ACM special interest group on data communication, Budapest, Hungary, pp 88–102
10. Mijumbi R, Hasija S, Davy S, Davy A, Jennings B, Boutaba R (2017) Topology-aware prediction of virtual network function resource requirements. IEEE Trans Netw Serv Manag 14(1):106–120
11. Chen Y, Wu J (2020) Flow scheduling of service chain processing in a NFV-based network. IEEE Trans Netw Sci Eng 8(1):389–399
12. Attaoui W, Sabir E, Elbiaze H, Sadik M (2020) Combined latency-aware and resource-effective virtual network function placement. In: Proc. of IEEE Canadian conference on electrical and computer engineering (CCECE), London, ON, Canada, pp 1–7
13. Zhang SQ, Zhang Q, Bannazadeh H, Leon-Garcia A (2015) Routing algorithms for network function virtualization enabled multicast topology on SDN. IEEE Trans Netw Serv Manag 12(4):580–594
14. Reyhanian N, Farmanbar H, Mohajer S, Luo ZQ (2020) Joint resource allocation and routing for service function chaining with in-subnetwork processing. In: Proc. of ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP), Barcelona, Spain, pp 4990–4994
15. Hawilo H, Jammal M, Shami A (2019) Network function virtualization-aware orchestrator for service function chaining placement in the cloud. IEEE J Sel Areas Commun 37(3):643–655
16. Wang S, Bi J, Wu JP, Vasilakos AV, Fan QL (2019) VNE-TD: A virtual network embedding algorithm based on temporal-difference learning. Comput Netw 161:251–263
17. Jin PP, Fei XC, Zhang QX, Liu F, Li B (2020) Latency-aware vnf chain deployment with efficient resource reuse at network edge. In: Proc. of IEEE conference on computer communications, Toronto, ON, Canada, pp 267–276
18. Ma WR, Beltran J, Pan ZL, Pan D, Pissinou N (2017) SDN-based traffic aware placement of NFV middleboxes. IEEE Trans Netw Serv Manag 14(3):528–542
19. Kumar G, Dukkipati N, Jang K, Wassel HMG, Wu X, Montazeri B, Wang YG, Springborn K, Alfeld C, Ryan M (2020) Swift: Delay is simple and effective for congestion control in the datacenter. In: Proc. of the annual conference of the ACM special interest group on data communication on the applications, virtual event, USA, pp 514–528
20. Zhang X, Hou WG, Guo L, Zhang QH, Guo PX, Li RJ (2020) Joint optimization of latency monitoring and traffic scheduling in software defined heterogeneous networks. Mob Netw Appl 25(1):102–113
21. Bogle J, Bhatia N, Ghobadi M, Menache I, Bjørner N, Valadarsky A, Schapira M (2019) TEAVAR: striking the right utilization-availability balance in WAN traffic engineering. In: Proc. of the 2019 conference of the ACM special interest group on data communication, Beijing, China, pp 29–43
22. Lyu Q, Zhu H (2019) Optimizing network latency with fast path assignment for incoming flows. Int J Comput Inf Eng 13(9):465–471
23. Kar B, Wu EHK, Lin YD (2017) Energy cost optimization in dynamic placement of virtualized network function chains. IEEE Trans Netw Serv Manag 15(1):372–386
24. Khoshkholghi MA, Khan MG, Noghani KA, Taheri J, Bhamare D, Kassler A, Xiang ZZ, Deng SG, Yang XX (2020) Service function chain placement for joint cost and latency optimization. Mob Netw Appl 25:2191–2205
25. Li DF, Hong PL, Xue KP, Pei JN (2018) Virtual network function placement considering resource optimization and SFC requests in cloud datacenter. IEEE Trans Parallel Distrib Syst 29(7):1664–1677
26. Qazi Z, Tu CC, Miao R, Chiang L, Sekar V, Yu ML (2013) Practical and incremental convergence between sdn and middleboxes, Open Network Summit, Santa Clara, CA, pp 1–13
27. Manias DM, Jammal M, Hawilo H, Shami A, Heidari P, Larabi A, Brunner R (2019) Machine learning for performance-aware virtual network function placement. In: Proc. of IEEE global communications conference (GLOBECOM), Waikoloa, HI, USA, pp 1–6
28. Pei JN, Hong PL, Pan M, Liu JQ, Zhou JS (2020) Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks. IEEE J Sel Areas in Commun 38(2):263–278
29. Nam H, Kim KH, Schulzrinne H (2016) QoE matters more than QoS: Why people stop watching cat videos. In: Proc. of the 35th annual IEEE international conference on computer communications, San Francisco, CA, USA, pp 1–9
30. N Bozkurt I, Aguirre A, Chandrasekaran B, Godfrey PB, Laughlin G, Maggs A, Singla B (2017) Why is the internet so slow?! In: Proc. of international conference on passive and active network measurement, Sydney, Australia, pp 173–187
31. Shaikh A, Rexford J, Shin KG (1999) Load-sensitive routing of long-lived IP flows. ACM SIGCOMM Comput Commun Rev 29(4):215–226
32. Truong-Huu T, Gurusamy M, Girisankar ST (2017) Dynamic flow scheduling with uncertain flow duration in optical data centers. IEEE Access 5:11200–11214
33. Antunes N, Pipiras V (2015) Sampling and censoring in estimation of flow distributions. In: Proc. of the 2015 IEEE international conference on communications (ICC), London, UK, pp 5865–5871
34. Chen L, Lingys J, Chen K, Liu F (2018) Auto: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. In: Proc. of the 2018 conference of the ACM special interest group on data communication, Budapest, Hungary, pp 191–205
35. Chen AY, Jin Y, Cao J, Li LE (2020) Tracking long duration flows in network traffic. In: Proc. of the 2010 IEEE international conference on computer communications, San Diego, CA, USA, pp 1–5
36. Poupart P, Chen ZT, Jaini P, Fung F, Susanto H, Geng YH, Chen L, Chen K, Jin H (2016) Online flow size prediction for improved network routing. In: Proc. of the 24th IEEE international conference on network protocols (ICNP), Singapore, pp 1–6

37. Liu WX, Cai J, Wang Y, Chen QC, Zeng JQ (2020) Fine-grained flow classification using deep learning for software defined data center networks. J Netw Comput Appl 168:1–16
38. Durner R, Kellerer W (2020) Network function offloading through classification of elephant flows. IEEE Trans Netw Serv Manag 17(2):807–820
39. Wang X, Shi WQ, Xiang Y, Li J (2015) Efficient network security policy enforcement with policy space analysis. IEEE/ACM Trans Netw 24(5):2926–2938
40. Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer Math 1(1):269–271
41. Watkins CJ, Dayan P (1992) Q-learning. Mach Learn 8(3-4):279–292
42. Cormen TH, Leiserson CE, Rivest RL, Stein C (2009) Introduction to algorithms. MIT Press, Cambridge
43. Leskovec J, Krevl A (2014) SNAP: Stanford network analysis project. https://snap.stanford.edu
44. Munir A, Qazi IA, Uzmi ZA, Mushtaq A, Ismail SN, Iqbal MS, Khan B (2013) Minimizing flow completion times in data centers. In: Proc. of the 2013 IEEE international conference on computer communications, Turin, Italy, pp 2157–2165

Affiliations

Qing Lyu[1] · Yonghang Zhou[2] · Qilin Fan[3] · Yongqiang Lyu[4] · Xi Zheng[5] · Guangquan Xu[6] · Jun Li[1]

Qing Lyu
lvq16@mail.tsinghua.org.cn

Yonghang Zhou
yonghangzhou@163.com

Yongqiang Lyu
luyq@tsinghua.edu.cn

Xi Zheng
james.zheng@mq.edu.au

Guangquan Xu
losin@tju.edu.cn

Jun Li
junl@tsinghua.edu.cn

1 The Department of Automation, Tsinghua University, Beijing, 100084, China

2 The Department of Software Engineering, Xiamen University-Malaysia Campus, Sepang, 43900, Malaysia

3 The School of Big Data and Software Engineering, Chongqing University, Chongqing, 401331, China

4 Beijing National Research Centre for Information Science and Technology (BNRist), Tsinghua University, Beijing, 100084, China

5 The Department of Computing, Macquarie University, Sydney, 2109, Australia

6 The Tianjin Key Laboratory of Advanced Networking (TANK), College of Intelligence and Computing, Tianjin University, Tianjin, 300350, China