Peer-to-Peer Networking and Applications (2025) 18:0
https://doi.org/10.1007/s12083-024-01861-1

Reinforcement Q-learning enabled energy-efficient service function chain provisioning in multi-domain networks

Zhiying Wang[1] · Guanhua Huang[1] · Gang Sun[1] · Hongfang Yu[1] · Jian Sun[1]

Received: 21 December 2023 / Accepted: 21 September 2024

Abstract
Network function virtualization (NFV) technology is an efficient way to address the increasing difficulty of provisioning and managing network services. However, NFV-related service function chaining (SFC) deployment in multi-domain networks remains challenging, and there is still room for performance improvement. This paper investigates many heuristic algorithms in the same field and proposes a new method for dynamic SFC deployment in a multi-domain network. In our study, we combine a heuristic algorithm with reinforcement learning and divide the complex problem into several parts. This algorithm efficiently gives the SFC deployment scheme in the multi-domain network with subdomain privacy protection requirements and considers the energy savings of the multi-domain networks. Compared with the existing approach, the proposed algorithm has superiorities in terms of deployment success ratio, deployment profit, time efficiency, and energy consumption.

Keywords Network function virtualization · Service function chaining · Energy-saving · Network service deployment · Reinforcement Q-learning

1 Introduction

Network function virtualization [1] involves programming the functions of dedicated network equipment, such as gateways and firewalls, by virtualization technologies in a network that supports the deployment of virtual machines, which virtualize as a virtual machine (VM) or container (Docker). Some network services that are associated with the hardware can be deployed in a software manner to achieve the same effect [2].

With the technology of network function virtualization, virtualized network functions (VNFs) [3] can generally be employed to represent the network services that are deployed on the nodes of the virtual function network topology. An SFC [4] is formed when several VNFs in a specific order are connected by a path to form a whole set of services. This technology reduces the hardware cost and operation cost [5, 6] but has requirements for computing resources, bandwidth resources, and robustness of the VNF [7].

After investigating the service requirements based on user requirements, the information for SFC deployment, such as user access point, data acquisition point, and required service function type, required resources (i.e., CPU, memory, and bandwidth), can be obtained [8]. A deployment scheme that satisfies both the requirements of users and the interests of service providers is needed [9]. VNF is deployed on the nodes of the physical network, and the data flow of a user is planned to form an SFC according to the scheme.

Since this technology was proposed, numerous studies have explored the virtualization of network functions [10, 11]. It remains a problem to determine an SFC beneficial to both users and providers, combining a service network with user demand. This type of problem, known as VNF deployment (virtualized network function-placement, VNF-P), is an NP-hard problem [12]. VNF deployment involves various optimization issues such as network structure [13], server performance [14] network transmission performance [15], network energy consumption [16], and multi-operator collaboration [17]. Solutions have evolved from the initial mathematical modeling of linear programming to the heuristic algorithm [18].

This article is part of the Topical Collection: *1- Track on Networking and Applications*
Guest Editor: Vojislav B. Misic

✉ Gang Sun
gangsun@uestc.edu.cn

[1] Key Laboratory of Optical Fiber Sensing and Communications (Ministry of Education), University of Electronic Science and Technology of China, Chengdu 611731, China

Published online: 23 December 2024

With advancements in computer performance, machine learning has gained the interest of researchers and has a role in prominent fields [19]. Machine learning is also introduced to the virtualization of network functions [20]. Determining how and where to use machine learning in this field for better results depends on researchers' continuous exploration and discovery.

2 Related work

SFC deployment problems include static SFC deployment and dynamic SFC deployment [21, 22]. In static SFC deployment, the deployment decision is one-time and the user's needs can be arranged in advance. An SFC will always exist in a network after deployment. Thus, the general goal is to increase the number of successful deployments. For example, the BSVR algorithm proposed in [23] mainly considers the load balance of the network and the number of SFCs that can be successfully deployed. In addition, the VNF set in [23] has sharing characteristics, and one VNF can be shared by multiple SFCs. For privacy protection, we assume that a VNF belongs to only one user in this paper.

In this paper, we study the dynamic SFC deployment [24], which is close to reality. User requirements come in sequence over a period of time. When a requirement arrives, an SFC must be deployed, and the deployment order of SFCs cannot be exchanged. The deployed SFC in the network will have an online time limit. When the online time is over, the SFC will be withdrawn, and the occupied network resources will be released.

The SFC deployment of a multi-domain network is more complicated. A multi-domain network is composed of multiple single-domain networks connected by top-domain links. Every single domain network may belong to different managers or different areas. Differences exist in the domain bandwidth resources and top-domain bandwidth resources. To satisfy a user requirement, several VNFs may need to be distributed in different single domains. The access nodes of a few top-domain paths will also affect the direction of an SFC in each domain. Refs. [25, 26] have studied cross-domain SFC orchestration. The authors model the SFC deployment problem in the multi-domain network to solve the optimization, then a heuristic algorithm is added to the algorithm to improve the performance. The authors of Ref [27]. propose the Cooperative Multi-agent Reinforcement Learning (CMRL) based algorithm for multi-domain SFC routing problem. The authors of Ref [28]. design a novel DRL framework based on the enhanced deep deterministic policy gradient (E-DDPG) for the efficient SFC embedding in the dynamic and complex cloud network scenarios. In [29], information is shared among subdomains and centralized control is used to achieve the purpose of multi-domain cooperative deployment of an SFC. However, this method is not conducive to multiservice providers or multi-region cooperation, and the private information of a subdomain may be leaked.

To resemble reality, privacy protection for every single-domain network may be required in the multi-domain network. The information between each single-domain network is not interoperable. Some information is not uploaded to the general control platform but is managed by the single-domain network's control platform. This scenario with privacy protection is more similar to the future network with the collaboration of multiple service providers. The authors of Ref. [30] investigate the collaboration framework of security services in the multi-domain network, which can negotiate not only the tasks to be launched or unloaded but also the duration of collaboration and the number of resources for each task to achieve multi-domain collaboration. In [31], satellite and ground networks are explored. Each ground single-domain network is operated cooperatively using satellite master control. Compared with the simple non-cooperative top-domain path calculation method, this method has less mapping delay, and operators can reasonably distribute the computational load without providing topology information. Ref. [32] is one of the few studies of the energy-saving of SFC deployment in multi-domain networks with the promise of protecting the privacy of subdomains. In the algorithm, the boundary nodes of top-domain communication are exposed through each subdomain to form an abstract network. The SFC path is gradually detailed and has an excellent energy-saving effect and time efficiency, but the success rate of deployment and the profitability of the service provider are not mentioned. Ref. [28] introduces an innovative DRL framework utilizing the enhanced deep deterministic policy gradient (E-DDPG) to effectively handle SFC embedding in dynamic and complex cloud network environments. Ref. [33] presents a deep learning model that integrates a multitask regression layer on top of graph neural networks to predict the future resource needs of each VNF instance.

The algorithm proposed in this paper combines reinforcement learning with heuristic algorithms and not only takes advantage of the swift decision-making speed of Q-learning but also employs heuristic algorithms to lessen the difficulty of some problems. The main contributions of this paper are as follows:

1. **Hierarchical Reinforcement Learning Framework:** We introduced a framework specifically designed for dynamic SFC deployment in multi-domain networks, innovatively partitioning the provisioning challenge into two hierarchical levels: top-level domain networks and sub-domain networks.
2. **Privacy-Aware Decision Making:** Our method distinctively embeds privacy into the reinforcement learning

process for multi-domain networks, pioneering independent decision-making within subdomains through our hierarchical model.
3. **Energy-Efficient SFC Provisioning:** Distinguishing itself, our algorithm prioritizes energy efficiency alongside deployment success and profitability within the RL framework.
4. **Adaptive Learning for Dynamic Networks:** Our method stands out for its adaptive learning capabilities, designed to evolve with the changing dynamics of network conditions and demands. Meanwhile, the algorithm avoids the slow training speed caused by the complicated Q-learning training model and provides a new method for solving this kind of problem.

3 Problem description and modeling

In this section, we study the deployment of dynamic SFC in a multi-domain network. A node's computing resources and channel bandwidth resources in the network are limited. SFC requests arrive sequentially, and after working for a while, the SFCs expire and release the allocated resources. The privacy of data within every single domain in the network is protected. Our goal is to improve the profit of operators and the success rate of SFC deployment.

There are many servers in a multi-domain network. Energy saving is one of the optimization goals of the algorithm. Turning off unserved machines can reduce costs and thus increase the service providers' profit.

In this multi-domain network scenario, composed of multiple single domains, also referred to as subdomains, each node is a general-purpose server with a unique capacity of computing resources. When the SFC passes, it starts up to provide function deployment or forwarding services. Otherwise, it remains off. The links in a multi-domain network are grouped into top-level domain links and sub-domain links. The top-level domain links and sub-domain links have different bandwidth resource capacities. Due to their strategic position as the few links connecting different domains, top-level domain links play a pivotal role in determining the efficiency and success of SFC deployment. Their limited capacity directly impacts the overall network performance, making them a critical factor that demands special attention in our study. Understanding and optimizing the utilization of these top-level domain links is essential for achieving a seamless and effective deployment of SFCs across the entire multi-domain network.

In this network scenario, there are always new user requirements arriving. Requirements include the source and destination node, duration, and specific virtual network functions. To meet more user requirements, the SFC must be withdrawn from the network when its offline time reaches. According to the actual situation, we need to consider data privacy protection for each domain. Therefore, some data in the single domain will not be reported to the general control platform and can only be processed by the sub-control platforms of the subdomain, making the problem more real and complex.

The objective is to effectively deploy an SFC in a multi-domain network, improve the success rate of SFC deployment, maximize the profits of service providers, shorten the decision time, meet all constraints including intra-domain privacy protection, and thus make the network more energy efficient.

3.1 Multi-domain network model with virtualization functions

The multi-domain network is composed of multiple single domains. In reality, every node within a single domain is a server or server group. Each node will have sub-domain links but only the nodes of borders have top-domain links.

The topology of the whole network is $G = (V, E)$, where V is the set of nodes and E is the set of edges between connected nodes. $v \in V$ is a specific node, and I_v is the computing resource of the node v. $e \in E$ is a specific edge in the topology, which may be a top-domain link or a sub-domain link, and B_e is the bandwidth resource of the link e. Both computing resources and bandwidth resources are limited and may be exhausted by SFC deployments. Nodes that possess greater computational resources and bandwidth are more suitable to serve as the nodes of borders.

The single-domain network is G_n^{inter}, where n is the number of subdomains. The top-domain links set is E^{top}, and the set of sub-domain links is E^{inter}. The elements in the sets are e^{top} and e_n^{inter}. E includes sub-domain links in n subdomains and top-domain links between subdomains.

For all nodes in the network, all kinds of VNFs can be deployed as long as sufficient compute resources are available. However, due to limited computing resources determined by the combination of CPU and memory, the total computing resources required by a node to deploy VNFs cannot exceed the computing resource capacity of that node, as expressed in Formula (1). The maximum capacity of the node's computing resources is I_max_v. I_v is computing resource consumption.

$$I_v \leq I\_max_v if v \in V \tag{1}$$

Bandwidth resources of network links are also limited, and the capacities of the two kinds of links are different. The maximum capacity of the sub-domain link is $B\_max_e^{inter}$, and that of the top-domain link is $B\_max_e^{top}$. The limited bandwidth resources in the links are shown as Formulas (2) and (3).

$$B_e \le B\_max_e^{top} ife \in E^{top} \tag{2}$$

$$B_e \le B\_max_e^{inter} ife \in E^{inter} \tag{3}$$

The positions and numbers of the nodes and paths in each single-domain network are also defined for security protection. The positions, numbers, and computing resources of nodes and the bandwidth resources of the sub-domain paths cannot be exposed. Only some fuzzy calculated values can be externally displayed for the linkage among multiple domains.

The energy consumption in the network is also considered. In this section, only the energy consumption of the server is taken into account; it is composed of boot consumption $Energy_{idle}$ and VNF load energy consumption $Energy_{vnf}$, as shown in Formula (4).

$$Energy_{total} = Energy_{idle} + Energy_{vnf} \tag{4}$$

The boot energy consumption is determined by whether the server is turned on or off. When no online SFC passes the node, the node is in a shutdown state and the energy consumption is zero. The load energy consumption of a node depends on the VNFs deployed on it.

3.2 Cross-domain service functional chain demand model

SFC deployment provides a solution according to the users' requirements. RE is the set of requirements in the multi-domain virtualization functional network$RE = \{S_i^{node}, S_i^{domain}, d_i^{node}, d_i^{domain}, P_i, r_i, t_i\}$. s_i^{node} is the user's access node, and s_i^{domain} is the user's access domain. The two values are combined as the starting node of the SFC to be deployed, which is given by the user and cannot be changed. A Boolean variable $x1$ indicates whether the SFC deployed according to the requirements satisfies this condition. $first\_node_i$ is the first node of the SFC, and the variable $x1$ is obtained by Formula (5).

$$x1 = \begin{cases} 1 if first\_node_i = s_i^{node}, s_i^{node} \in s_i^{domain} \\ 0 else \end{cases} \tag{5}$$

Another node that cannot be changed is a termination node in SFC. d_i^{node} is the data supply node, and d_i^{domain} is the domain where the node is located to provide the data needed by the user. The Boolean variable $x2$ indicates whether the deployed SFC satisfies the condition according to the requirements: 1 is satisfied, and 0 is not satisfied. $last\_node_i$ is the last node of the SFC, and the variable $x2$ is obtained by Formula (6).

$$x2 = \begin{cases} 1 if last\_node_i = d_i^{node}, d_i^{node} \in d_i^{domain} \\ 0 else \end{cases} \tag{6}$$

P_i is an ordered array that represents several VNFs required in this SFC, where the elements are the VNFs that need to be deployed in order. $length_{SFC\_VNF_i}$ is the total number of VNFs that need to be deployed for this particular configuration of the SFC. $length_{P_i}$ is the number of VNFs required to deploy the SFC from the start node to the end node. The Boolean variable $x3$ indicates whether the deployed SFC satisfies this condition. SFC_VNF_i is the ordered VNF list from the starting node to the ending node of the SFC to be deployed, and the variable $x3$ is obtained by Formula (7).

$$x3 = \begin{cases} 1 if length_{SFC\_VNF_i} = length_{P_i} \\ and SFC\_VNF_i[j] = P_i[j], \\ 0 \le j < length_{P_i} \\ 0 else \end{cases} \tag{7}$$

To facilitate the calculation of profits, r_i is the income generated by the successful deployment of the SFC. r_i is generally proportional to the number of VNFs deployed in the SFC. r_i is calculated by Formula (8), where $m > 1$ is determined by the size of the network and β is the unit value.

$$r_i = length_{P_i} * m * \beta \tag{8}$$

Different VNFs have a different consumption of computing resources, which is represented by ω_j. The cost of setting up the VNF also differs depending on the amount of computing resource consumption, and the cost coefficient is o_j. The computing resource consumption is proportional to the cost. The cost of deploying an SFC is $cost_i$, as shown in Formula (9).

$$cost_i = \sum_{j \in P_i} o_j * \omega_j * \beta \tag{9}$$

According to the cost and benefit of an SFC, the profit of deploying the SFC can be obtained. The profit is $profit_i$ as expressed by Formula (10).

$$profit_i = r_i - cost_i \tag{10}$$

When the profit is positive, the SFC is worth deploying, which is consistent with reality. The Boolean variable $x4$ indicates whether the SFC satisfies the condition, a value of 1 is satisfied, and a value of 0 is not satisfied. The variable $x4$ is obtained by Formula (11).

$$x4 = \begin{cases} 1 if profit_i \ge 0 \\ 0 else \end{cases} \tag{11}$$

After successful deployment, a special parameter for dynamic SFC deployment is the duration of the SFC t_i.

3.3 Dynamic service function chain deployment model

Different from static SFC deployment, the requirements arrive with an irregular time interval over a certain time range in dynamic SFC deployment. We cannot choose the order in which users' requirements arrive. For a requirement, a decision should be made based on whether it can be deployed immediately. A decision is more complicated than a static SFC deployment problem, which can rearrange the deployment order. The arrival time of demand generally conforms to the Poisson distribution.

In dynamic SFC deployment, the deployment requirements are accompanied by the online duration t_i, which means that the deployed SFC will withdraw from the network after a certain time and return the occupied network resources, behaving closer to the real network. After this specified duration, the deployed SFC automatically withdraws from the network, releasing the occupied resources back to the pool. This mechanism closely mimics the behavior of real-world networks, where services are not static but are active for a limited period. Different SFCs have different durations, and they are randomly generated within a range and conform to a normal distribution.

According to the user requirements, a successful SFC deployment needs to satisfy the following points: the source and destination node are correct; the required VNF is successfully deployed; a continuous path from the start node to the end node exists; and the benefit value is greater than the cost value. Otherwise, this deployment fails. The Boolean variable $success_i$ indicates whether the SFC can be successfully deployed: 1 is successful, and 0 is unsuccessful. The calculation Formula (12) is expressed as follows:

$$success_i = x1 \cap x2 \cap x3 \cap x4 \tag{12}$$

In this way, the total profit of the entire deployment process $profit_{total}$ can be calculated:

$$profit_{total} = \sum_{i \in RE} success_i * profit_i \tag{13}$$

The goal of dynamic SFC deployment, which is similar to static SFC deployment, is to deploy as many SFCs as possible. Different network structures, the fixed order of arrival of requirements, and limited bandwidth resources and computing resources, etc. will affect the deployment, overall SFC deployment success ratio and network service provider profits.

4 Algorithm design

SFC deployment is more complicated in a multi-domain network. Our research deploys SFC with clear paths between domains and conserves energy of the whole network with the promise of protecting the privacy of each single domain. The algorithm combines Q-Learning and a heuristic algorithm, layers the multi-domain network and separately trains them. The privacy data in a single domain is fuzzified and then reported. The selection of a path is optimized by the energy-saving scoring module.

4.1 Q-learning model optimization

The problem in this paper conforms to the Markovian decision process but is slightly different from the general reinforcement learning problem. A limited number of decisions is required in this problem. Therefore, this paper combines the Q-learning algorithm with this problem and optimizes the model. In this way, we not only harness the adaptability and decision-making capabilities of reinforcement learning but also navigate around its inherent challenges, offering an innovative solution for dynamic SFC deployment. This approach enables us to amalgamate the strengths of RL with the robustness of metaheuristic algorithms like GA and PSO, effectively mitigating their respective limitations.

The improved Q-learning algorithm in this paper improves the Q matrix and training algorithm. We transform the two-dimensional matrix into a five-dimensional matrix, which has the nodes *now_h*,*now_node*, *action_node*, *end_node*, and *h*. *now_h* is the number of nodes that have passed in the current state, that is, the current chain length. *now_node* is the node where the agent is in the current state. *action_node* is the possible next node, which is the adjacent node of the current node. *end_node* is the node to which the SFC will eventually reach. *h* is the minimum number of passing nodes that are required in the deployment requirements and are generally related to the number of VNFs to be deployed. Only if the length of the alternative path is greater than the number of VNFs to be deployed, will a discussion of it should be employed as the output result be necessary.

Specifically, the single subscript state *s* of the original Q matrix are represented by the four subscripts *now_h* ,*now_node*, *end_node* and *h*, while action *a* is represented by the *action_node*. The current state is determined by four subscripts, with the exception of *action_node*. The action with the large value was selected for execution.

The advantage of replacing the two-dimensional matrix with a five-dimensional matrix is that observing the current state is easier. The information on the network topology is closely related to each state. If a single index is used to represent the state, various states should be numbered, which increases the workload. The five-dimensional matrix can clearly observe the state information, and the intervention can be more timely and more adaptive to catastrophes of the dynamic network. On the other hand, this matrix form can divide all states into several independent parts, and this

division is conducive to parallel programming and can further accelerate the training process.

4.2 General algorithm design

In this section, we introduce the Q-learning multi-domain SFC deployment algorithm for dynamic SFC deployment. The proposed algorithm consists of two stages:

As shown in Fig. 1, the demand enters the hierarchical network reinforcement learning routing module. The main task of this module is to perform routing in a multi-domain network and output a set of alternative routes.

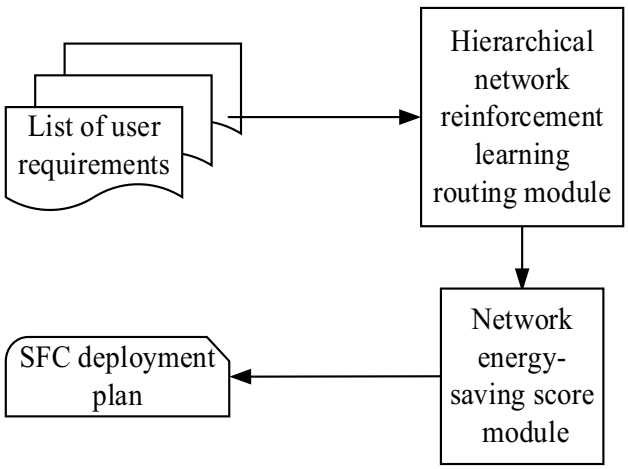


Fig. 1 General algorithm framework

Different from the completely transparent information in a single domain network, a multi-domain network has subdomain privacy protection. The network must be layered, the privacy information should be separated, and then the training and use of reinforcement learning should be separately conducted.

The main task of the network energy-saving score module is to select the best path from the alternative path set given by the previous module for deployment to keep the energy consumption at a low level. The information cannot be directly reported because the privacy of the subdomain is to be protected. The information needs to be fuzzified, uploaded to the general control platform, and scored. The deployment plan is then output.

The next two sections will detail the two modules and the algorithm steps. Some of the variable definitions in the algorithm are shown in Table 1.

4.3 Hierarchical network reinforcement learning routing module

The main function of the hierarchical network reinforcement learning routing module is to output a set of alternative paths that satisfy the user's SFC requirements. The following content consists of three parts: to introduce network layering, the training and decision algorithm of reinforcement learning for path selection between two domains, and the path selection algorithm in each subdomain. Combined with the previous research results, the optimized Q-learning algorithm is employed as the reinforcement learning algorithm.

Table 1 Key notations in this paper

Notation	Definition
G	The multi-domain network topology
G_n^{inter}	The internal topology of the subdomain
G^{top}	The abstract of the top network
V	A list of all nodes in the network topology
V_i	A list of nodes adjacent to node i in the network
V_{access}	A list of cross-domain nodes
h_{max}	The maximum length of the SFC in the subdomain
h_{min}	The minimum length of the SFC in the subdomain
Q^{top}	A four-dimensional memory matrix, stores the recommended values for actions in each state
Q_n^{inter}	A five-dimensional memory matrix, stores the recommended values for actions in each state
R^{top}	A four-dimensional reward matrix, stores the reward or penalty values for some combination of states and actions
R_n^{inter}	A five-dimensional memory matrix, stores the reward or penalty values for some combination of states and actions
h	The length of the chain formed in the current state
RE	The list of SFC requests
PA^{top}	The set of paths that match the starting and ending nodes only in the top network
PA_k^{inter}	The set of paths that match the starting and ending nodes only in the subdomain network
$num\_vnf_i^k$	Number of VNF nodes that can be deployed in the i-th path in the k-th subdomain
ONL	List of online SFCs

4.4 Multi-domain network layering

Multi-domain network has many subdomains, which may belong to the same network service provider, and can have unified management. Thus, this network can be regarded as a superlarge single-domain network, and the previous single-domain algorithm is suitable with a slight modification. However, each subdomain may also belong to different Internet service providers or different regions. In this case, each subdomain has a unique control and management platform. The data and resources in the network with privacy protection requirements will not continue to be uploaded, which is a new challenge for services collaboration in multi-domain networks.

Multi-domain network has subdomain privacy protection demands while provisioning an SFC request. The locations of the nodes in the subdomain, connection of channels and remaining resources are inconvenient for the entire network to share, which hinders the ability to establish a unified reinforcement learning model and can only be separately addressed. A hierarchical network model and the respective reinforcement learning models need to be established. By some numerical processing, the blurring information of each subdomain can be collected and reported for further decision-making.

The network topology G is composed of the n subdomain G_n^{inter} and top-domain paths. The subdomain is regarded as an abstract node, the top-domain links are considered the links between these abstract nodes, and the abstract network topology is the top network G^{top}. For the general control platform of the top network, the available information includes the location and resources of the links and the information reported by the virtual nodes. Fig. 2

A simple example is shown in Fig. 3, which includes an original multi-domain network with 5 subdomains. The black nodes in each subdomain are nodes that can communicate between two domains, and the white nodes are in the domain. Seven top-domain links exist between each domain and are indicated by the dashed lines. Abstract and layering this multi-domain network, the top network of the network and each subdomain network are shown on the right side of Fig. 3. The structure of the top network is similar to the single-domain network; the difference is that the value reported by the abstract node is the fuzzy value within the original subdomain to which it points, and this fuzzy value is calculated by the formula for multi-domain collaborative work to protect the privacy of the subdomain. The value has no practical meaning after leaving the SFC deployment problem.

After the multi-domain network is layered, the top network G^{top} and n subdomain networks G_n^{inter} will exist. The processing of these networks is separately discussed in the following sections. The processing consists of two parts:

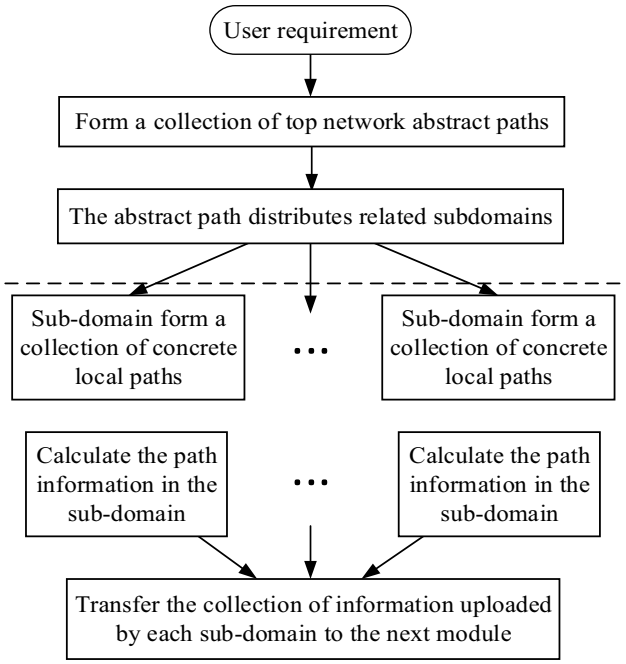


Fig. 2 Process of implementation in routing module

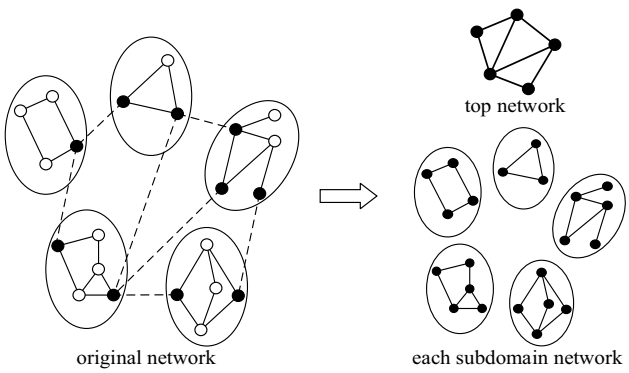


Fig. 3 Process of implementation in routing module

the top-domain Q-learning training and decision algorithm of the top network and the sub-domain Q-learning training and decision algorithm of the subdomain network.

For the top-domain Q-learning training and decision algorithm of the top network, we must determine the form of the Q matrix Q^{top}. This matrix is a four-dimensional memory matrix with four subscripts: *now_h*,*now_node*, *action_node*, and *end_node*. The minimum hop counts in routing of the top network may not have to be considered. Because the source and destination nodes of the SFC may be in two connected subdomains, h is 1, and some values in the algorithm do not have to be skipped.

For the sub-domain Q-learning training and decision algorithm of the subdomain network, the Q matrix can refer

to the path selection section of SFC deployment in a single-domain network. The memory matrix of the subdomain G_n^{inter}, Q_n^{inter} is a five-dimensional memory matrix with five subscripts: *now_h*,*now_node*, *action_node*, *end_node* and *h*.

4.5 Top-domain Q-learning training algorithm

The goal of Algorithm 1 is to train the Q^{top} matrix in the top network. For the R^{top} reward matrix, set *now_node* and *end_node* to 1000, which indicates the reward for completing the route search, and set the remaining elements to 0.

Different from the single domain Q-learning training algorithm, Algorithm 1 does not specify the length interval limit of the searched path. Due to the nature of the top network, the size of the nodes will not be too large; so the algorithm changes to remember all the paths from the shortest two nodes to the longest path without looping. Formula (1) is further concretized to obtain Formula (14) for Q-learning training.

$$Q(s_i, a_i) = 0.8\left(r + \max_{a_i'} Q(s_i', a_i')\right) \tag{14}$$

Algorithm 1 Top-domain Q-learning training

```
Initialize the Q^top as all 0 matrix according to G^top;
Initialize R^top;
For node v ∈V in G^top, do:
    Initialize list chain, add node v;
    Call function Find_top_way (Q^top,R^top,G^top, chain);
End For
```

Function: Find_top_way (Q, R, G, $chain$)

```
Let v0 = the last element in chain;
Let temporary list chain_tmp = chain;
While chain_tmp does not contain all nodes and v0's adjacent nodes are not
all in chain_tmp, do:
    For adjacent node v2 ∈ V_v0 of v0, do:
        If node v2 is not in chain_tmp, then:
            Add v2 into chain_tmp;
            Call Find_ top_way (Q^top,R^top,G^top, chain_tmp);
            For element in chain_tmp, in reverse order do:
                Write the information from chain_tmp into matrix Q ac-
                cording to Formula (15);
            End For
        End If
    End For
End While
```

4.6 Top-domain Q-learning training algorithm

When the training finishes, the Q^{top} generation file is saved by the system. In the decision stage, Q^{top} is used to output a certain number of abstract alternative paths PA^{top}, which satisfies the user's SFC cross-domain requirements, represents the passed subdomains and top-domain links in the original network. The purpose of this algorithm is to form these abstract paths and send them to the relevant subdomain's sub-control platform for subsequent generation of the next layer path.

In Algorithm 2, part of the data is sent to the network energy-saving score module, that is, the set of alternative abstract paths in the top network. Subsequently, the

sub-control platform of each subdomain decides the path in the specific domain and reports the fuzzy value after calculation. Only after the two parts of data are combined, can the path score be compared.

Algorithm 2 Top-domain Q-learning decision

```
Read the trained Q^top;
Read the list of user requirements RE;
For each user requirement re in RE, do:
    Get some alternative paths correspond to the starting and ending fields
    from Q^top and add them to PA^top;
    For each abstract path pa in PA^top, do:
        If top-domain path of pa is unobstructed, then:
            Send the related information of pa and re to the relevant sub-
            do main of the path;
        End If
    End for
    If PA^top is empty, then:
        The SFC deployment of user' requirement re fails;
    End If
    Send PA^top to the network energy-saving score module for further data
    collection;
End For
```

4.7 Sub-domain Q-learning training algorithm

Since each subdomain has a unique sub-control platform, the algorithm can be executed in parallel to improve the efficiency. The algorithm is slightly different from the single-domain Q-learning training algorithm. For the reward matrix R_n^{inter}, among the elements with the same index of *now_node* and *end_node*, *end_node* represents the elements in the domain used to connect to other domains are set to 1000, which is the reward for completing the route search, and the remaining elements are set to 0.

The convergent matrix Q can be applied in each subdomain for path selection. Two different points exist between Algorithm 3 and the single-domain Q-learning training algorithm. First, the nodes in the subdomain that communicate with other domains must be the starting and ending nodes of a path. During the training process, the workload of other nodes as the end node to a loop is reduced. Second, when writing to memory, the reverse writing is to take the nodes in the subdomain that are employed for connecting with other domains as the end nodes, while the positive writing is to take the nodes as the starting nodes and form two path memories at a time.

Algorithm 3 Sub-domain Q-learning training

Initialize the Q_n^{inter} as all 0 matrix according to G_n^{inter};
Initialize R_n^{inter};
Set the value of h_{max};
Let $h = 0$;
For cross-domain node $v \in V_{access}$ in G_n^{inter}, **do:**
    Initialize list $chain$, add node v;
    Call function **Find_double_way** (Q_n^{inter}, R_n^{inter}, G_n^{inter}, h_{max}, h, $chain$);
End For

***Function*:** Find_double_way (Q, R, G, h_{max}, h, $chain$)

Let $v0$ = the last element in $chain$;
$h = h + 1$;
Let temporary list $chain\_tmp = chain$;
While $h <= h_{max}$, **do:**
    For adjacent node $v2 \in V_{v0}$ of $v0$, **do:**
        If node $v2$ is not in $chain\_tmp$, **then:**
            Add $v2$ into $chain\_tmp$;
            Call function **Find_double_way** (Q, R, G, h_{max}, h, $chain\_tmp$);
            For element in $chain\_tmp$, in reverse order **do:**
                Write the information from $chain\_tmp$ into matrix Q according to Formula (15);
            End For
            For element in $chain\_tmp$, in positive order **do:**
                Write the information of $chain\_tmp$ into matrix Q according to formula (15);
            End For
        End If
    End For
End While

4.8 Sub-domain Q-learning decision algorithm

In the decision-making stage, the network general control platform will send *pa* and *re* to subdomains that each abstract path *pa* in the alternative path set of the top network PA^{top} has passed, and then generate the paths in each subdomain and perform the preliminary processing.

The starting and ending nodes of the path in the subdomain are determined by *pa* and *re*. If the domain is a start–end domain, one start–end node is a node in the domain, and the other start–end node is a border node that connects other domains. If the domain is a subdomain of an intermediate path, the start–end nodes are two border nodes. The number of generated paths in the subdomain should be limited to facilitate subsequent processing and improve the efficiency of multi-domain SFC deployment. The optimization goal of the algorithm in this section is to save the energy of the network server. Thus, Formulas (15) and (17) are used to screen paths, and the best few paths are reserved. Formula (15) ensures that the bandwidth resources of the link in the domain are sufficient for deploying the SFC.

$$B_{max_e^{inter}} - B_e > B_{SFC}, e \in SFC \tag{15}$$

Formula (16) calculates the approximate energy consumption using the path, including the basic energy consumption of the new boot node and the maximum energy consumption

required to deploy the VNF. In Formula (16), num_{new} is the number of unpowered nodes that exist on the path, $Energy_{idle}$ is the basic energy consumption of each node that needs to be turned on for the path, which contains energy expenditure for a machine to transition from an off state to an operational state, $num_{node\_on\_pa_i^k}$ is the number of nodes on the path, $Energy_{vnf}$ is the energy consumption of the VNF that requires the most computing resources in the deployment requirements. k is the domain number and i is the path number.

$$Energe_i^k = num_{new} * Energy_{idle} + Energy_{vnf} * num_{node\_on\_pa_i^k} \tag{16}$$

For subsequent paths, the contents of the report should be fuzzy to protect the privacy of the subdomain. We use Formula (17) to abstract the situation in the subdomain and redistribute the values for the subsequent report. The value calculated by Formula (18) is only relevant to this problem and does not have an alternate meaning.

$$\varphi_i^k = \frac{Energe_i^k * 100}{1000} \tag{17}$$

Algorithm 4 Sub-domain Q-learning decision

Read the trained Q_n^{inter} of subdomain G_n^{inter};
Read the user requirement re and path set PA^{top};
For each abstract path pa in PA^{top}, **do:**
    Get all subdomains G_n^{inter} that pa has passed through;
    For all the sub-control platforms of each subdomain G_k^{inter} in G_n^{inter} run concurrently, **do:**
        Generate the path set PA_k^{inter} that satisfies the requirements of the source and destination nodes in the subdomain;
        Use Formulas (16) and (17) to screen pa_i^k from PA_k^{inter};
        Use Formula (18) to calculate and report the blurring value φ_i^k of each path from the abridged PA_k^{inter} and sort them ascendingly;
        If the nodes in pa_i^k can deploy the VNF that requires the most resources in re, **then:**
            Record the number of nodes that satisfy the condition as num_vnf_i^k;
        End If
    End for
    If the screened PA_k^{inter} is empty, **then**:
        The SFC deployment of abstract path pa fails;
        continue;
    End If
    Each subdomain G_k^{inter} send the ordered φ_i^k and num_vnf_i^k in PA_k^{inter} to the next module;
End For

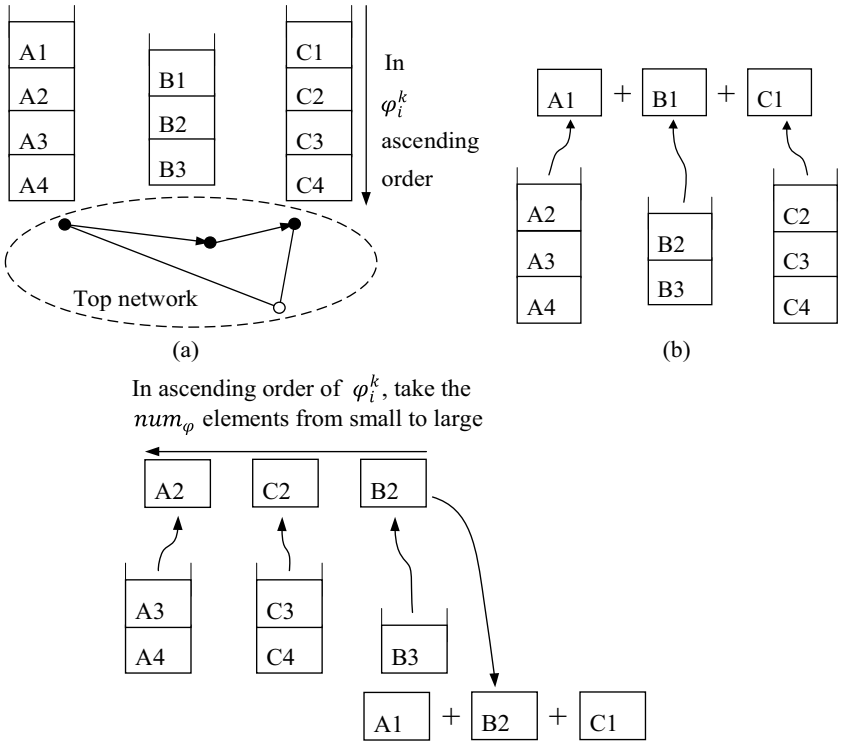


Fig. 4 Process of path combination in stack structure (a) Subdomains in the abstract path report information φ_i^k and $num\_vnf_i^k$; (b) Pop up the top element of the stack to form a specific path, and output the scheme if VNF can be deployed; (c) If the SFC in (b) is not deployable, try the alternate path

Algorithm 4 is the execution process after each *pa* in PA^{top} is distributed to the subdomain. When Algorithm 4 records the number of deployable VNF nodes $num\_vnf_i^k$, the VNF that consumes the most resources in the list of user requirements *re* is employed as the standard, which reduces the probability of deployment failures due to insufficient resources. Note that φ_i^k and $num\_vnf_i^k$ of the alternative route set PA_k^{inter} in the domain are already arranged in ascending order of φ_i^k to facilitate subsequent path selection.

4.9 Network energy-saving score module

Because of the privacy protection of the subdomains in the multi-domain network, the network energy-saving score module can only process the information of user demand *re*, the abstract path of the top network PA^{top}, as well as φ_i^k and num_vnf_i^k of the alternative path set PA_k^{inter} in each subdomain.

The process of forming an abstract path into a concrete path is shown in Fig. 3. For the user demand *re*, when the total control platform of the top network gives a set of alternative abstract paths PA^{top}, considering one of the abstract paths *pa* as an example, it has already considered whether the bandwidth of the top-domain path allows deployment. The energy consumption of the path between two nodes is not discussed here.

Figure 4 shows the selection rules of the specific path formed by the abstract path; a decision about the process is made by using the stack structure: (*i*) Each subdomain of the abstract path *pa* will submit the φ_i^k and $num\_vnf_i^k$ of the set of feasible paths in the domain. This information is applied as a stack formed by the elements. φ_i^k are pushed onto the stack in ascending order. (*ii*) The top element of the stack represents the most energy-efficient path. The combination of pa_i^k with the minimum φ_i^k in each subdomain is theoretically the most energy-efficient solution. Therefore, one element is popped from the top of each stack, and the paths represented by these elements form an SFC. However, whether VNF in the requirement can be deployed on this SFC should be considered by determining whether the sum of $num\_vnf_i^k$ in each element is greater than the number of VNFs in the user requirement. (*iii*) If the path in (*ii*) is not able to deploy all the VNF in the user requirements. To improve the success rate of deployment, we find the minimum φ_i^k in the φ_i^k with the second minimum subscript *i*, and replace part of the path in the original scheme. Thus, the top elements of the stacks are popped again for sorting. To further improve the deployment success ratio and reduce the workload, when looking for alternative paths of standby partial paths, among these sorted elements, the num_φ elements with the minimum φ_i^k can be directly reserved. num_φ is the path rollback value, which as an adjustable value, can be adjusted and optimized according to the specific situation.

After determining all the paths in the related subdomains and confirming that VNF can be deployed, the score of the whole SFC is given. $weight_{G_k^{inter}}$ is the weight factor of the subdomain, which increases to reduce the use of the subdomain. If no special requirements exist, they are set to 1. The score calculation Formula (18) is defined as follows:

$$score = \sum_{G_k^{inter} \in SFC} weight_{G_k^{inter}} * \varphi_i^k \tag{18}$$

The stack structure is utilized to generate the specific path, and Formula (18) is used to calculate the score. The algorithm of the network energy-saving score is defined as follows:

Algorithm 5 Network energy-saving score

Read the related information from G^{top};
Read user requirement *re* and alternative path list PA^{top};
Read φ_i^k and $num\_vnf_i{}^k$ of alternative path set in subdomain PA_k^{inter} of the abstract path *pa* in PA^{top};
For *pa* in PA^{top}, **do:**
    Generate the whole SFC and calculate score of *pa;*
End For
Choose path with the lowest score from PA^{top};
Record SFC's online starting time $t_{start-re}$ and calculate SFC's evacuation time;
Add the SFC information to the online SFC list *ONL;*
Change occupancy record of the associated resource in the network topology;
If any SFC in *ONL* has reached the evacuation time, **then:**
    Return the occupied resource of topology;
End If

5 Simulation results and analysis

5.1 Simulation environment

The simulation result is the average value of data obtained by multiple tests to ensure the accuracy of the experiment. Since the results are normally distributed, the average basically represents the most frequently occurring result value. The simulation program of the experiment was run on a computer with a CPU of 2.5 GHz Intel Core i5-2450 m and 6 GB memory. The operating system was Windows 10. The algorithm model and simulation code are written with Java.

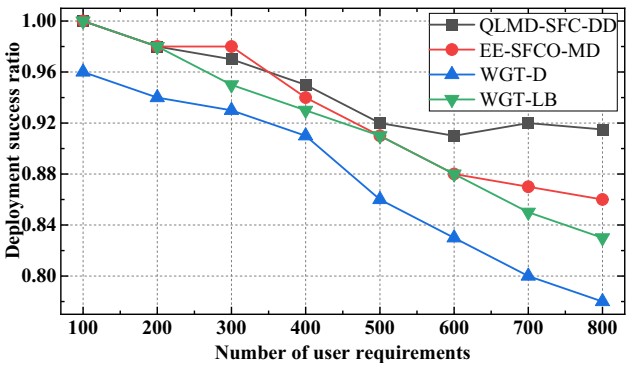


Fig. 5 Comparison of deployment success ratio with a different number of user requirements

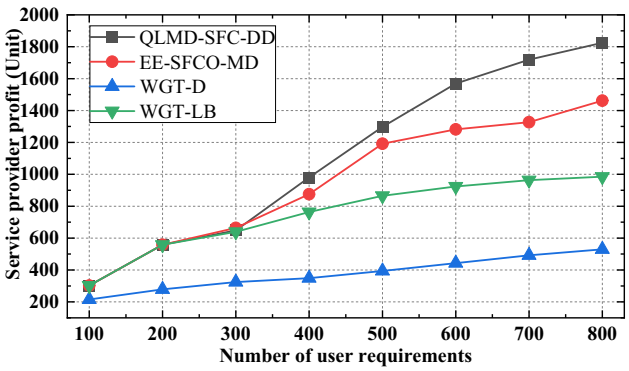


Fig. 6 Comparison of total profit of network operators for a different number of user requirements

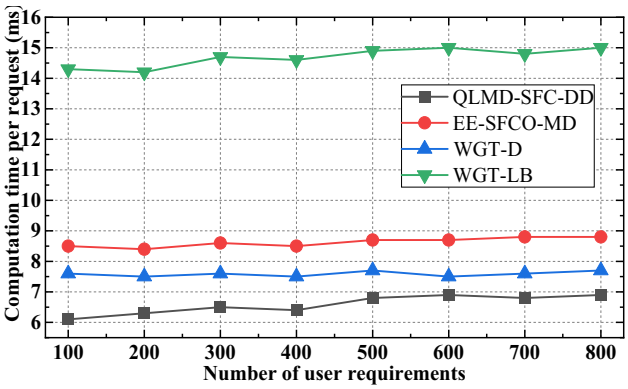


Fig. 7 Comparison of SFC deployment decision time for a different number of user requirements

The multi-domain network has the randomly generated network topology structure, which has 6 subdomains, and each subdomain has 10 to 20 nodes and 30 to 60 edges. Each node in this topology acts as a server, which not only provides functions of routing and forwarding but can also deploy multiple types of VNFs. The server supports all 7 types of VNFs when computing resources are sufficient.

To test the task of dynamic SFC deployment, the channel bandwidth resources in the network are limited, and the total capacity of the node computing resources is 100 units. Different.

A VNF of type j consumes ω_j units of computational resources. The value of ω_j ranges from 1 to 7. Due to the characteristics of network function virtualization technology, the deployed VNF is set to only serve one user to protect users' privacy.

The bandwidth capacity of the top-domain link is 15 units, and that of each sub-domain link is 25 units. Each deployed SFC occupies 1 unit of bandwidth resources in its path. The basic power consumption of a server is set to 140 units. Since the computing resources that each VNF consumes is different, the load energy consumption ranges

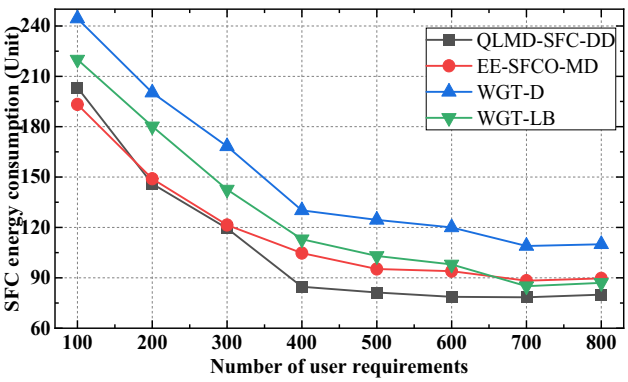


Fig. 8 Comparison of energy consumption of single SFC for a different number of user requirements

from 1 to 7 units, which is proportional to the consumption of the computing resources. To resemble reality, the online time of each user request follows a uniform distribution, and the arrival time follows a Poisson distribution, which ensures the randomness of the arrival and withdrawal of the SFC deployment request. In addition, the number of VNFs in the SFC requested by each user is normally distributed between 3 and 6. In the simulation, this paper used 6 sub-domains to connect in different ways, creating three different topological structures. Perform 10 experiments in each topology to obtain the average value as the result output.

The proposed algorithm is compared with those in [31] and [34], which also address the problem of dynamic SFC deployment in multi-domain networks. In the simulation comparison, the algorithm in [31] is referred to as EE-SFCO-MD, the algorithms in [34] are referred to as WGT-D and WGT-LB, and the algorithm proposed in this paper is referred to as QLMD-SFC-DD.

5.2 Performance index

For the dynamic SFC deployment in the virtualized multi-domain network, the following indexes are used in the simulation to evaluate the performance: deployment success ratio, profit of network operator, average deployment decision time and average power consumption.

The deployment success ratio A is the ratio of the number of SFCs that are successfully deployed on the network to the number of all incoming requests. The calculation of A is obtained from Formula (19).

$$A = \frac{Number_{successfullydeployedSFC}}{Number_{inputservicerequests}} \tag{19}$$

The profit of the network operator $profit_{total}$ is the total profit of the network service provider after receiving the input SFC request. The calculation Formula (20) is expressed as follows:

$$profit_{total} = \sum_{i \in RE} success_i * (r_i - cost_i) \tag{20}$$

The average deployment decision time C for an SFC request reflects the decision time required to deploy each SFC; it is calculated by dividing the total simulation time

Table 2 Parameters of simulation

	$num_\varphi = 0$	$num_\varphi = 3$	$num_\varphi = 6$
λ^{top}	0.2—0.4	0.2—0.4	0.2—0.4
λ^{inter}	0.4—0.6	0.4—0.6	0.4—0.6
num_φ	0	3	6

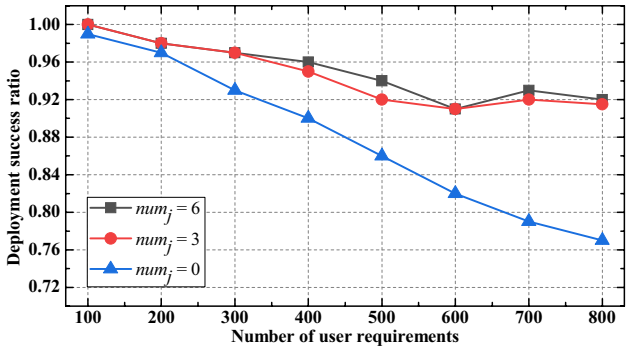


Fig. 9 Comparison of the deployment success ratio of three different values of num_φ

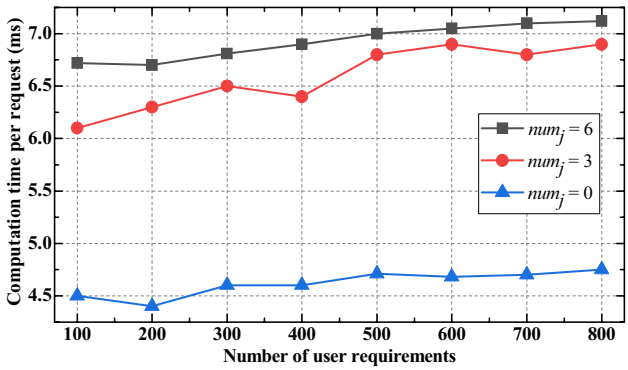


Fig. 10 Comparison of the deployment decision time of three different value of num_φ

by the total number of input SFC requests. The calculation in shown in Formula (21).

$$C = \frac{Totalrunningtime}{Number_{inputservicerequests}} \tag{21}$$

The average power consumption E is the sum of the power consumption divided by the total number of successfully deployed SFCSs. The calculation formula of the average power consumption of each deployed SFC is shown in Formula (22).

$$E = \frac{\sum_{i \in RE} success_i * Energy_i^{SFC}}{Number_{successfullydeployedSFC}} \tag{22}$$

5.3 Power calculation module

The simulation results consist of two parts: the comparison and performance analysis between the proposed algorithm and the other algorithm, and the demonstration and analysis of the impact of some adjustable parameters on the algorithm's performance.

5.4 Deployment success ratio

Figure 5 shows the comparison of the deployment success ratio of the four algorithms. When the number of SFC requests is less than 200, the deployment success ratio of the QLMD-SFC-DD, EE-SFCO-MD, and WGT-LB algorithms can almost reach 100% because of the abundant network resources. The WGT-D algorithm deploys sub-chains to minimize latency, which can lead to uneven deployment and ultimately cause deployment failures in hotspot areas. With the increasing number of requests, the success rate starts to decrease. However, the difference is minimal before 500, which indicates that the network has not reached the near saturation of resource bearing. Starting from 600, the QLMD-SFC-DD algorithm has a large number of alternative paths, and the deployment success ratio is approximately 91%. The deployment success ratio of EE-SFCO-MD is gradually reduced to approximately 87% and the deployment success ratio of WGT-LB is gradually reduced to approximately 83% due to the shortage of resources in the network. The findings concluded that the deployment success ratio of the QLMD-SFC-DD algorithm is similar to the comparison algorithm when the number of SFC requests is small. When the amount of SFC requests is large, the proposed algorithm has a higher deployment success ratio than the comparison algorithm for the environmental conditions set by the simulation.

5.5 Profit of network operator

Figure 6 shows the comparison results of the network operators' profits of the four algorithms. When the number of SFC requests is less than 300, the profits of the QLMD-SFC-DD, EE-SFCO-MD, and WGT-LB algorithms are similar and increase with an increase in the number of deployments. The WGT-D algorithm's low deployment success ratio affects the profit of the network operator, making it lower than that of the other three algorithms. When the deployments range from 300 to 500, as a result of the limited number of network resources, the algorithms' way of routing differ and have different costs and profits. After the number of SFC requests has risen to more than 500, due to the different deployment success ratios, the profit of network operators also substantially differ. When the number of user requests reaches 800, the profit of the proposed algorithm is increased by 23.6% more than that of the other algorithms.

5.6 Average deployment decision time

Figure 7 shows the comparison results of the deployment decision time of the four algorithms. All algorithms have excellent performance in terms of decision time and exhibit minimal rise with an increasing number of SFC requests. The WGT-LB algorithm deploys sub-chains with the

objective of load balancing, which results in a longer SFC deployment decision time due to its global impact. However, as a heuristic algorithm, this time remains within an acceptable range. After our proposed algorithm is optimized, the time is shorter than that of the other algorithms. When the number of user requests is 100, the deployment decision time of the proposed algorithm is 39.3% shorter than that of the other algorithm.

5.7 Energy consumption optimization

Figure 8 shows the comparison results of the energy consumption optimization of the four algorithms. When the number of SFC requests is low, the number of SFCs that need a large number of new startups accounts for a large proportion of successfully deployed SFCs. Thus, the average energy consumption is relatively high but gradually declines as the number of SFC requests increases. The network becomes stable in the later stage of deployment; thus, the average energy consumption of each SFC also tends to be stable. With the promise that the energy consumption of SFC that the EE-SFCO-MD algorithm deployed is better than other algorithms, we can conclude that the algorithm in this paper is not worse than the other algorithms and the main goal can be achieved while taking into account the energy-saving optimization of the network.

Summarizing all the comparison experiments, we conclude that the proposed algorithm outperforms the comparison algorithms in terms of success ratio, profit, and decision time while ensuring comparable energy consumption.

5.8 Influence of the adjustable parameters

The algorithm has three kinds of dynamically adjustable parameters. The first two kinds are λ^{top} and λ^{inter}. They are used to regulate the utilization rate of paths output by the Q matrix of the top network and each subdomain network, respectively. Another adjustable parameter is num_{φ}, which is used to control the number of alternative partial paths after filling the specific path of the abstract path fails for the first time.

For λ^{top} and λ^{inter}, the higher is the utilization rate, the larger is the number of paths output by the Q matrix. In this problem, the number of abstract paths of the top network and the number of local paths of each subdomain will increase, which will improve the deployment success ratio to a certain extent but will increase the running time of the algorithm. We choose to increase the number of output local paths in the subdomain, so λ^{inter} chooses to use λ_{Middle} and there are not many interdomain paths, so λ^{top} chooses to use λ_{Low} to speed up the output time. Therefore, the parameters λ^{top} and λ^{inter} are fixed, and the parameter num_{φ} is changed. We observe the trend of the deployment success ratio and the deployment decision time. The values of the parameters are set as shown in Table 2.

Figure 9 is the simulation result of the deployment success ratio for the three numerical values of num_{φ}. When num_{φ} is 0, the deployment of the SFC path is abandoned after the first-generation failure, which yields a lower deployment success ratio than the other two cases, and the gap increases with the number of requests. In the subsequent stage, when the network is close to saturation, due to resource constraints, large numbers of user requirements cannot be deployed. When num_{φ} is 3 or 6, the difference between the two is not large, because after providing 3 substitute local paths, the SFC path can be successfully generated. Thus, in the case of providing 6 substitute paths, usually, a deployment scheme can be successfully obtained by the first three paths. Different topologies may have a unique num_{φ}, which is determined by the connection of the topological distribution in the subdomain. To attempt the value from small to large, we can obtain the set value of the parameter when the deployment success ratio does not substantially change. In the condition that the success rate of deployment is not different, the profit of deployment is also the same.

Figure 10 is the simulation result of the deployment decision time for three numerical values of num_{φ}. When num_{φ} is 0, the selection and ordering of substitute local paths are completely omitted; thus, the time is reduced. However, when num_{φ} is 6, 3 more substituting local paths need to be sorted than when the value is 3, and the time will slightly increase. When num_{φ} is 0, a short time is needed, and the deployment success ratio is lower, which will reduce the profit. When the value of num_{φ} increases, the decision time will also increase. To optimize the performance of the algorithm, we assign the minimum value of num_{φ} that does not substantially change the deployment success ratio.

In the comparison simulation between the proposed algorithm and other algorithms, the proposed algorithm sets num_{φ} to 3 to ensure the deployment success ratio, improve profits, and reduce the deployment decision time as much as possible.

The average energy consumption of the deployed SFC was compared to evaluate the energy-saving performance. The parameters λ^{top} and λ^{inter} have a great impact on energy consumption, and more path choices may provide more energy-efficient routing. The parameter num_{φ}, with the condition that the other two parameters are fixed, does not substantially contribute to saving energy. When num_{φ} is 0, it is the most energy-efficient scheme to deploy but has the lowest deployment success ratio. Assume that the top-of-stack scheme is not employed and these alternate paths are utilized. When num_{φ} is greater than 0 before the stack structure converts the abstract path into a specific path, the data in the stack have been sorted in ascending order according to the energy consumption, and the greater num_{φ} will not

have a better energy-saving effect than the original scheme formed by the top of the stack.

6 Conclusion

This paper investigates the dynamic SFC deployment in multi-domain networks with privacy protection. We optimize the Q-learning algorithm in the dynamic network and design a new deployment decision algorithm that combines reinforcement learning and a heuristic algorithm.

To protect the privacy of each subdomain in the multi-domain network, the information in the subdomain cannot be uploaded directly, which further enlarges the difficulty of SFC deployment. We also add the module of network energy-saving to reduce energy consumption and operating cost. The algorithm layered the network and separated the top network and subdomain networks. The reinforcement learning module was employed to select alternative abstract paths in the top network and was then distributed to each subdomain to form specific parts of the abstract path in parallel. By uploading fuzzy information, the network energy-saving scoring module scores each alternative path and finally obtains the deployment scheme. In the simulation experiment, the proposed algorithm is compared with the benchmark algorithm. We find that the algorithm in this paper has achieved excellent results in terms of the decision time, network energy savings, deployment success ratio, and deployment benefits.

The future work directions and ideas consist of two parts, optimizing the algorithm and expanding the problem domain:

(1) Study some adjustable parameters that can affect the results in the algorithm to explore whether different optimal schemes exist in different situations, and further optimize the performance of the algorithm. With the enhancement of the computing power and the popularization of large-scale cluster computing, the distributed or parallel operation mode of the algorithm can be explored.

(2) According to the current progress, the future work of the virtualized functional network will be carried out on the independent resource management of the service network [35], virtual machine migration of VNF deployment [36], further energy-saving operation of servers and channels in the network [37], and decentralization of resource allocation controller [38].

Author contributions Zhiying Wang wrote the main part of the manuscript. Guanhua Huang developed the model and performed experiments and performed the experiments. All authors read and approved the final manuscript.

Funding This research was partially supported by the National Key Research and Development Program of China (2019YFB1802800).

Data availability Not applicable.

Declarations

Ethics approval This work does not contain any studies with human participants or animals performed by any of the authors.

Consent to participate Not applicable.

Consent for publication Not applicable.

Conflicts of interest The authors declare no competing interests.

References

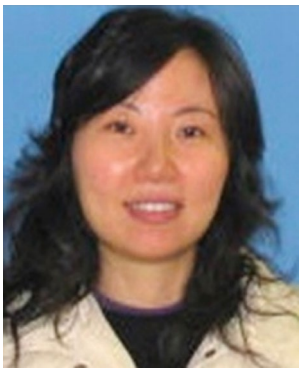
1. Adoga UH, Pezaros PD (2022) Network function virtualization and service function chaining frameworks: A comprehensive review of requirements, objectives, implementations, and open research challenges. Future Internet 14(2):59
2. Balakrishnan H, Banerjee S, Cidon I et al (2021) Revitalizing the public internet by making it extensible[J]. ACM SIGCOMM Computer Commun Rev 51(2):18–24
3. Zhao D, Luo L, Yu H et al (2021) Security-SLA-guaranteed service function chaining deployment in cloud-fog computing networks. Clust Comput 24:2479–2494
4. Sun G, Xu Z, Yu H et al (2021) Dynamic network function provisioning to enable network in box for industrial applications. IEEE Trans Industr Inf 17(10):7155–7164
5. Sun J, Zhang Y, Liu F et al (2022) A survey on the placement of virtual network functions. J Netw Comput Appl 202:1–37
6. Yi B, Wang X, Li K et al (2018) A comprehensive survey of Network Function Virtualization. Comput Netw 133:212–262
7. Sun G, Zhou R, Sun J et al (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
8. Moens H, Turck F (2016) Customizable function chains: Managing service chain variability in hybrid NFV networks. IEEE Trans Netw Serv Manage 13(4):711–724
9. Fan Q, Pan P, Li X et al (2022) DRL-D: revenue-aware online service function chain deployment via deep reinforcement learning. IEEE Trans Netw Serv Manag 19(14):4531–4545
10. Varasteh A, Madiwalar B, Van AB et al (2021) Holu: power-aware and delay-constrained VNF placement and chaining[J]. IEEE Trans Netw Serv Manage 18(2):1524–1539
11. Jia Z, Sheng M, Li J et al (2021) VNF-based service provision in software defined LEO satellite networks[J]. IEEE Trans Wireless Commun 20(9):6139–6153
12. Xuan H, You L, Liu Z et al (2021) HS-MOEA/D: An oriented algorithm for delay and reliability VNF-SC deployment. Security and Commun Networks 2021:5538931
13. Zhang B, Fan Q, Zhang X et al (2024) A survey of VNF forwarding graph embedding in B5G/6G networks. Wirel Netw 30(5):3735–3758
14. Montazerolghaem A (2021) Software-defined load-balanced data center: design, implementation and performance analysis[J]. Clust Comput 24(2):591–610

15. Wang Y et al (2022) Energy-efficient method based on dynamic topology switching and reliability in SDNs. IEEE Trans Sustain Comput 7(2):427–440
16. Liu Y, Ran J, Hu H et al (2021) Energy-efficient virtual network function reconfiguration strategy based on short-term resources requirement prediction[J]. Electronics 10(18):2287
17. Subramanya T, Riggio R (2021) Centralized and federated learning for predictive VNF autoscaling in multi-domain 5G networks and beyond[J]. IEEE Trans Netw Serv Manage 18(1):63–78
18. Sun G, Li Y, Liao D et al (2018) Service function chain orchestration across multiple domains: A full mesh aggregation approach. IEEE Trans Netw Serv Manage 15(3):1175–1191
19. Bertolini M, Mezzogori D, Neroni M et al (2021) Machine learning for industrial applications: A comprehensive literature review[J]. Expert Syst Appl 175:114820
20. Moradi M, Ahmadi M, Nikbazm R (2022) Comparison of machine learning techniques for VNF resource requirements prediction in NFV. J Netw Syst Manage 30(1):1–29
21. Wu Y, Zhou J (2021) Dynamic service function chaining orchestration in a multi-domain: A heuristic approach based on SRv6[J]. Sensors 21(19):6563
22. Xu L, Hu H, Liu Y et al (2023) SFCSim: a network function virtualization resource allocation simulation platform. Clust Comput 26:423–436
23. Li D, Lan JL, Wang P et al (2018) Joint service function chain deploying and path selection for bandwidth saving and VNF reuse. Int J Commun Syst 31(6):e3523
24. Sun G, Liao D, Zhao D et al (2018) Live migration for multiple correlated virtual machines in cloud-based data centers. IEEE Trans Serv Comput 11(2):279–291
25. Melo M, Sargento S, Killat U et al (2015) Optimal virtual network embedding: Energy aware formulation. Comput Netw 91:184–195
26. Sun G, Anand V, Liao D et al (2015) Power-efficient provisioning for online virtual network requests in cloud-based data centers. IEEE Syst J 9(2):427–441
27. Zhao D, Lu Y, Li X, Li Z, Liu Y (2022) Cross-Domain Service Function Chain Routing: Multiagent Reinforcement Learning Approaches. IEEE Trans Circuits Syst II Express Briefs 69(12):4754–4758
28. Liu Y, Zhang J et al (2024) Service function chain embedding meets machine learning: deep reinforcement learning approach. IEEE Trans Netw Serv Manag 21(3):3465–3481
29. Dietrich D, Abujoda A, Rizk A et al (2017) Multi-provider service chain embedding with nestor. IEEE Trans Netw Serv Manage 14(1):91–105
30. Migault D, Simplicio MA, Barros BM et al (2018) A framework for enabling security services collaboration across multiple domains. Comput Electr Eng 69:224–239
31. Li G, Zhou H, Feng B et al (2018) Horizontal-based orchestration for multi-domain SFC in SDN/NFV-enabled satellite/terrestrial networks. China Communications 15(5):87–101
32. Sun G, Li Y, Yu H et al (2019) Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. Futur Gener Comput Syst 91:347–360
33. Zhang J, Liu Y, Li Z, Lu Y (2023) Forecast-Assisted Service Function Chain Dynamic Deployment for SDN/NFV-Enabled Cloud Management Systems. IEEE Syst J 17(3):4371–4382
34. Sun G, Li Y, Liao D, Chang V (2018) Service Function Chain Orchestration Across Multiple Domains: A Full Mesh Aggregation Approach. IEEE Trans Netw Serv Manage 15(3):1175–1191
35. Khatiri A, Mirjalily G, Luo QZ (2022) Balanced resource allocation for VNF service chain provisioning in inter-datacenter elastic optical networks. Comput Netw 203:108717
36. Zhang Q, Liu F, Zeng C (2021) Online adaptive interference-aware VNF deployment and migration for 5G network slice. IEEE/ACM Trans Networking 29(5):2115–2128
37. Tong Z, Cai J, Mei J et al (2022) Dynamic energy-saving offloading strategy guided by lyapunov optimization for IoT devices. IEEE Internet of Things J 9(20):19903–19915
38. Santos HRF, Ferreira NT, Mattos FMD et al (2022) An efficient and decentralized fuzzy reinforcement learning bandwidth controller for multitenant data centers. J Netw Syst Manage 30(4):1–24

Zhiying Wang is a student at the University of Electronic Science and Technology of China. Her research interests include network function virtualization, mobile edge computing, deep reinforcement learning, Internet of Things, and unmanned aerial vehicle.

Guanhua Huang is a student at the University of Electronic Science and Technology of China. His research interests encompass network function virtualization, deep reinforcement learning, the Internet of Things, and service function chaining.

Gang Sun (Senior Member, IEEE) is a full professor of computer science with the University of Electronic Science and Technology of China. His research interests include network virtualization, cloud computing, high performance computing, parallel and distributed systems, ubiquitous/pervasive computing, IoT, and blockchain. He has edited special issues at top journals, such as the IEEE Vehicular Technology Magazine, IEEE Internet of Things Magazine, Future Generation Computer Systems and Multimedia Tool and Applications. He has served as reviewers of the IEEE Journal on Selected Area of Communications, IEEE/ACM Transactions on Networking, IEEE Transactions on Industrial Informatics, IEEE Wireless Communications Magazine, IEEE Transactions on Network and Service Management, IEEE Communications Letters, Information Fusion, Future Generation Computer Systems, Journal of Network and Computer Applications.



Hongfang Yu (Senior Member, IEEE) received the BS degree from the Xidian University, in 1996, and the MS and PhD degrees from the University of Electronic Science and Technology of China (UESTC), China, in 1999 and 2006, respectively. She is currently a full professor with the Key Laboratory of Optical Fiber Sensing and Communications (Ministry of Education), UESTC, and the vice dean with the School of Information and Communication Engineering, UESTC. Her research interests include data center networking, network function virtualization, software-defined networking, blockchain technology, and federated learning. She is an area editor of the IEEE Internet of Things Journal, a technical editor of the IEEE Network, an associate editor of the IEEE Communications Surveys & Tutorials, and Digital Communications and Networks.

Jian Sun received the BS and MS degrees from the University of Electronic Science and Technology of China, in 1992 and 1998, respectively. He is currently a senior engineer at the University of Electronic Science and Technology of China, enjoying the benefits of a professor-level position. His research interests include computer communication networks, mobile internet, internet of things technology, network and information security, big data and data mining, and machine learning and artificial intelligence.