Hindawi
Security and Communication Networks
Volume 2024, Article ID 9895616, 1 page
https://doi.org/10.1155/2024/9895616

Retraction

Retracted: Deep Reinforcement Learning-Based Algorithm for VNF-SC Deployment

Security and Communication Networks

Received 8 January 2024; Accepted 8 January 2024; Published 9 January 2024

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] J. Xuan, H. Yang, X. Zhao, X. Ma, and X. Yang, "Deep Reinforcement Learning-Based Algorithm for VNF-SC Deployment," *Security and Communication Networks*, vol. 2021, Article ID 7398206, 11 pages, 2021.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 7398206, 11 pages
https://doi.org/10.1155/2021/7398206

Review Article

Deep Reinforcement Learning-Based Algorithm for VNF-SC Deployment

Junlei Xuan,[1] Huifang Yang ,[1] Xuelin Zhao,[2] Xingpo Ma,[2] and Xiaokai Yang[3]

[1]*School of Foreign Languages, Xinyang Normal University, Xinyang 464000, Henan, China*
[2]*School of Computer and Information Technology, Xinyang Normal University, Xinyang 464000, Henan, China*
[3]*Texas Precision Optics Inc, Houston 75220, Texas, USA*

Correspondence should be addressed to Huifang Yang; hfyang_fl@126.com

Received 24 July 2021; Accepted 24 September 2021; Published 6 October 2021

Academic Editor: Xingsi Xue

Network function virtualization (NFV) has the potential to lead to significant reductions in capital expenditure and can improve the flexibility of the network. Virtual network function (VNF) deployment problem will be one of key problems that need to be addressed in NFV. To solve the problem of routing and VNF deployment, an optimization model, which minimizes the maximum index of used frequency slots, the number of used frequency slots, and the number of initialized VNF, is established. In this optimization model, the dependency among the different VNFs is considered. In order to solve the service chain mapping problem of high dynamic virtual network, a new virtual network function service chain mapping algorithm PDQN-VNFSC was proposed by combining prediction algorithm and DQN (Deep Q-Network). Firstly, the real-time mapping of virtual network service chains is modeled into a partial observable Markov decision process. Then, the real-time mapping process of virtual network service chain is optimized by using global and long-term benefits. Finally, the service chain of virtual network function is mapped through the learning decision framework of offline learning and online deployment. The simulation results show that, compared with the existing algorithms, the proposed algorithm has a lower the maximum index of used frequency slots, the number of used frequency slots, and the number of initialized VNF.

1. Introduction

In the traditional network, in order to provide a variety of network services, operators need to deploy a large number of monitors, load balancers, firewalls, intrusion detection systems, and other different network functions. These network functions (NF) generally require specific devices to be physically deployed to realize, and the network data flow of some network functions that need to cross is called the network function chain [1–3]. Network function virtualization (NFV) is a technology that utilizes virtualization to separate network functions from dedicated hardware. Then, the virtual network function is mapped to the general server, switch, or memory to form the virtual network function (VNF) [4, 5]. This technology can not only reduce the cost of network construction and operation but also improve the flexibility of the network [6–8]. Therefore, the network data flow of some virtual network functions that need to go through is called VNF Service Chain (VNF-SC). Virtual network function configuration in the virtual network function chain is a key problem to be solved in the implementation of network function virtualization, and how to solve the problem of virtual network function configuration is a key to solve the problem of network function virtualization [9–11].

In recent decade years, there are large number works focusing on the virtual network function service chain deployment problem, such as [12, 13]. Zhu et al. [8], through calculating K alternative path data center deployment of virtual network function of the virtual network with required function of the longest common subsequence between sequence, determine the function of virtual network routing of service chain and virtual network function deployment plan, designed to maximize the network in the virtual network in the reuse rate. Considering that virtual network functions can be migrated, Tachun's team [14] proposed routing, deployment, and migration algorithm of virtual network function service chain based on rollback strategy, so as to minimize the bandwidth occupied by the rejected virtual network function service chain and the energy consumed in the network. Aiming at the problem of finding the best placement of service chain in distributed cloud environment, Mechtri et al. [15] proposed an algorithm of adjacency matrix eigendecomposition based on infrastructure topology diagram and requested virtual network function forwarding graph. Under the background of network function virtualization, the research on the resource scheduling scheme oriented to virtual network function service chain is mainly based on three kinds of methods: heuristic method [10, 16], optimization model method [17], and learning theory-based method [18]. Based on the heuristic method, the resource scheduling scheme can be obtained quickly, but it is easy to fall into local optimal. The method based on the optimization model can get the optimal scheduling scheme, but it is difficult in modeling and solving. With the development of high performance computing and deep neural network, the study of the method resource scheduling scheme based on learning theory has attracted the attention of many scholars. The basic idea of the method based on learning theory is to obtain the resource scheduling scheme under different network states through learning strategies. In order to get the approximate optimal mapping scheme of virtual network function service chain, Quang et al. [18] proposed a solution method based on reinforcement learning search in a large action space to get the optimal mapping scheme. In order to solve the deployment problem of virtual network functions in software-defined networks, the problem was modeled as 0-1 integer programming problem in [19], and a virtual network function deployment algorithm based on double Q network was proposed. In order to minimize the energy consumption in the network, Solozabal et al. [20] adopt the combinatorial optimization theory and deep reinforcement learning to carry out the virtual network function deployment algorithm. Baojia et al. [21] study the problem of virtual network function deployment and propose a deep reinforcement learning method based on AC(Actor-Critic), which can well obtain the deployment scheme of virtual network function according to the current network state.

In this paper, the problem VNFs deployment for VNF-SC is investigated. To solve the problem of routing and VNF deployment, an optimization model, which minimizes the maximum index of used frequency slots, the number of used frequency slots, and the number of initialized VNF, is established. In this optimization model, the dependency among the different VNFs is considered. In order to solve the service chain mapping problem of high dynamic virtual network, a new virtual network function service chain mapping algorithm PDQN-VNFSC was proposed by combining prediction algorithm and DQN (Deep Q-Network). Firstly, the real-time mapping of virtual network service chains is modeled into a partial observable Markov decision process. Then, the real-time mapping process of virtual network service chain is optimized by using global and long-term benefits. Finally, the service chain of virtual network function is mapped through the learning decision framework of offline learning and online deployment.

The rest of this paper is organized as follows. Section 2 gives the network architecture and the optimization model. To solve the optimization model effectively, we propose an improved brain storm optimization algorithm in Section 3. To evaluate the algorithm proposed, simulation experiments are conducted, and the experimental results are analyzed in Section 4. The paper is concluded with a summary in Section 5.

2. Problem Description and Mathematical Modeling

2.1. Problem Description. Undirected graph $G = (V, E)$ represents a network topology, where $V = \{V_1, V_2, \ldots, V_{N_V}\}$ and $E = \left\{l_{ij} | V_i, V_j \in V\right\}$ represent the set of network nodes and the set of network links in the network, respectively. N_V and N_E denote the number of nodes and links in the IoT network. Network nodes represent devices in the network, such as gateways, routers, and switches. These nodes only have the function of network forwarding, without the functions of monitor, load balancer, firewall, and intrusion detection system. Some of the network nodes in the network topology are connected to the data center, and some software can be deployed to implement the related functions. Therefore, the nodes in the network can be represented by a binary group $V_i = (i, \Omega_i)$, where $\Omega_i \neq 0$ denotes the number of data centers connected to the nodes; otherwise, $\Omega_i = 0$. N_{DC} denotes the number of data centers in the network. All virtual network functions can be realized in any data center. The set of virtual network functions is expressed as $\mathrm{VNF} = \left\{\mathrm{VNF}_1, \mathrm{VNF}_2, \ldots, \mathrm{VNF}_{N_{\mathrm{vnf}}}\right\}$, where N_{vnf} represents the number of virtual network functions. $E = \left\{l_{ij} | V_i, V_j \in V\right\}$ denotes the number of nodes in the network and $l_{ij} = 1$ denotes the link between network nodes V_i and V_j, otherwise $l_{ij} = l_{ij} = 0$. There are N_F frequency slots on each link and the numbered as $1, 2, \ldots, N_F$, respectively.

$R = \left\{R_1, R_2, \ldots, R_{N_R}\right\}$ represents a set (N_R denotes the number of virtual network function service chains (VNF-SC)) of virtual network function service chains, where $R_k = (s_k, d_k, \mathrm{VNF}_k^D, \mathrm{VNF}_k^I)$ represents the kth VNF-SC. s_k and d_k represent the source node and the destination node of the virtual network service chain R_k, respectively. VNF_k^D is the set of dependent virtual network functions required by the virtual network service chain R_k and is represented as $\mathrm{VNF}_k^D = \left\{\mathrm{VNF}_{k_1}^D, \mathrm{VNF}_{k_2}^D, \ldots, \mathrm{VNF}_{k_{N_D}}^D\right\}$, so we have $\mathrm{VNF}_k^D \subseteq \mathrm{VNF}$. The virtual network functions in VNF_k^D must be implemented in a fixed order in which the order is noncommutative, that is, if $i < j$, $\mathrm{VNF}_{k_i}^D$ need to be implemented before $\mathrm{VNF}_{k_j}^D$. VNF_k^I is the set of independent virtual network functions required by the virtual network service chain R_k and is represented as $\mathrm{VNF}_k^D =$

$\left\{\mathrm{VNF}^I_{k_1}, \mathrm{VNF}^I_{k_2}, \ldots, \mathrm{VNF}^I_{k_{N_I}}\right\}$. Similarly, we also have $\mathrm{VNF}^I_k \subseteq \mathrm{VNF}$. Different from the virtual network functions in VNF^D_k and the virtual network functions in VNF^I_k the set are implemented in an arbitrary order, and there is no interdependence between any two virtual network functions. $B^D_k = (b_k, B^D_k, B^I_k)$, where b_k represents the number of frequency slots required to be occupied when the virtual network function service chain R_k has not realized any virtual network functions, and $B^D_k = (b^D_{k_1}, b^D_{k_2}, \ldots, b^D_{k_{N_D}})$ represents the ratio of the number of frequency slots occupied by the virtual network function service chain R_k after realizing the corresponding dependent virtual network functions to the number occupied by the virtual network function when the virtual network function has not been realized. Similarly, $B^I_k = (b^I_{k_1}, b^I_{k_2}, \ldots, b^I_{k_{N_I}})$ represents the ratio of the number of frequency slots occupied by the virtual network function service chain R_k after the corresponding independent virtual network function is implemented to the number occupied by the virtual network function when the virtual network function is not implemented.

The routing of static virtual network function service chain and the configuration of virtual network function in elastic optical network between data centers can be summarized as follows. When a batch of virtual network function service chains arrive, how to choose the appropriate path for each virtual network function service chain, configure its required virtual network functions in the corresponding data center, and allocate appropriate frequency gap for them to achieve a certain objective optimization?

2.2. Mathematical Modeling. This paper aims to obtain a virtual network function chain routing, virtual network function configuration, and spectrum allocation scheme that can minimize the maximum frequency slots occupied in the network, the minimum frequency slots occupied in the network, and the minimum number of deployed virtual network functions in all data centers. In this paper, the weighted summation method is adopted to transform the three-objective optimization problem into a single-objective constrained optimization problem, and the optimization objective is normalized. Therefore, the optimization objective of the optimization model established in this paper can be expressed as

$$\min f = \min\left\{\frac{\alpha N^M_F}{N_F} + \frac{\beta N^U_F}{N_F N_E} + \frac{\gamma N^I_{\mathrm{vnf}}}{N_{DC} N_{\mathrm{vnf}}}\right\}, \qquad (1)$$

where N^M_F, N^U_F, and N^I_{vnf} are, respectively, the maximum frequency slots' number occupied in the network, the number of frequency slots occupied in the network, and the number of virtual network functions configured in the network. $\alpha, \beta,$ and γ are three weight coefficients; $\alpha \geq 0, \beta \geq 0, \gamma \geq 0,$ and $\alpha + \beta + \gamma = 1$. Since $N^M_F \leq N_F$, $N^U_F \leq N_F N_E$, and $N^I_{\mathrm{vnf}} \leq N_{DC} N_{\mathrm{vnf}}$, thus, we have $0 \leq f \leq 1$. Some conditions need to be satisfied in the routing of the virtual network function chain, the configuration of the virtual network function, and the spectrum allocation, that is, the constraint conditions are satisfied:

(a) Any virtual network function service chain $R_k\,(R_k \in R)$ can only occupy one path in the candidate path set, i.e.,

$$a\sum_{q=1}^{N_Q} \lambda^q_k = 1, \quad k = 1, 2, \ldots, N_R, \qquad (2)$$

where N_Q is the number of paths in the candidate path set of the virtual network function service chain $R_k\,(R_k \in R)$. If and only if the virtual network function service chain $R_k\,(R_k \in R)$ occupies the qth path in its candidate path set $Q_k = \left\{Q^1_k, Q^2_k, \ldots, Q^{N_Q}_k\right\}$, $\lambda^q_k = 1$, otherwise $\lambda^q_k = 0$.

(b) All the virtual network functions required can be realized on the data center on the occupied path of the service chain $R_k\,(R_k \in R)$; then,

$$\mathrm{VNF}^D_k \cup \mathrm{VNF}^I_k \subseteq \bigcup_{V_i \in V^q_k} \mathrm{VNF}^i_k, k = 1, 2, \ldots, N_R, \qquad (3)$$

where V^q_k is the set of nodes connected with data centers in the candidate path set of service chain of virtual network functions R_k and VNF^i_k is the set of virtual network functions realized on nodes V_i connected with data centers.

(c) The data center on the occupied path linked to the virtual network function service chain $R_k\,(R_k \in R)$ is able to satisfy all the required virtual network function dependencies, i.e.,

$$\mathrm{VNF}^D_{k_i} \in \bigcup_{V_i \in V^q_k\left({}^{\prime}_{t}\right)} \mathrm{VNF}^i_k, \quad \forall\left\langle \mathrm{VNF}^D_{k_t}, \mathrm{VNF}^D_{k_{'_t}}\right\rangle, \qquad (4)$$

where $\left\langle \mathrm{VNF}^D_{k_t}, \mathrm{VNF}^D_{k_{'_t}}\right\rangle$ represents two virtual network functions in the set of virtual network functions $\mathrm{VNF}^D_{k_t}$ and $\mathrm{VNF}^D_{k_{'_t}}$, and $\mathrm{VNF}^D_{k_{'_t}}$ are dependent on $\mathrm{VNF}^D_{k_t}$; that is, $\mathrm{VNF}^D_{k_t}$ must be implemented prior to $\mathrm{VNF}^D_{k_{'_t}}$. $V^q_k(t')$ represents the set of nodes (including $\mathrm{VNF}^D_{k_{'_t}}$ the nodes connected to the data center) in the path occupied by the service chain of the virtual network function, in which $\mathrm{VNF}^D_{k_{'_t}}$ are in front of the nodes connected to the data center.

(d) Any of the virtual network functions that need to be implemented by linking to the virtual network function service chain $R_k\,(R_k \in R)$ can only be configured on one data center; then,

$$\sum_{V_i \in V^q_k} \phi^i_{kt} = 1, \quad \forall \mathrm{VNF}_t \in \left(\mathrm{VNF}^D_k \bigcup \mathrm{VNF}^I_k\right). \qquad (5)$$

If and only if the virtual network function VNF_i required by the virtual network function service chain R_k is configured in the data center connected to the node V_t, $\phi^i_{kt} = 1$; otherwise, $\phi^i_{kt} = 0$.

Since the first fit strategy is intended to be used for spectrum allocation, the allocation schemes all satisfy the constraints such as frequency slot consistency and frequency slot continuity, so the formal description of the constraints required for spectrum allocation is no longer given. Based on the given objective function and constraint conditions, the optimization model established in this paper is as follows:

$$\begin{cases} \min \quad f = \min\left\{ \dfrac{\alpha N_F^M}{N_F} + \dfrac{\beta N_F^U}{N_F N_E} + \dfrac{\gamma N_{\mathrm{vnf}}^I}{N_{DC} N_{\mathrm{vnf}}} \right\} \\ \text{s.t.} \begin{array}{l} (a)\ \sum_{q=1}^{N_Q} \lambda_k^q = 1, \quad k = 1, 2, \ldots, N_R \\ (b)\ \mathrm{VNF}_k^D \cup \mathrm{VNF}_k^I \subseteq \bigcup_{V_i \in V_k^q} \mathrm{VNF}_k^i, R_k \in R \\ (c)\ \mathrm{VNF}_{k_i}^D \in \bigcup_{V_i \in V_k^q(t')} \mathrm{VNF}_k^i, \forall \left\langle \mathrm{VNF}_{k_t}^D, \mathrm{VNF}_{k_t}^D \right\rangle \\ (d)\ \sum_{V_i \in V_k^q} \phi_{\mathrm{kt}}^i = 1, \quad \forall \mathrm{VNF}_t \in \left(\mathrm{VNF}_k^D \cup \mathrm{VNF}_k^I \right). \end{array} \end{cases} \tag{6}$$

Analysis found that the established optimization model is a discrete, nonconvex optimization model; the traditional derivative information such as the optimization method is not applicable to solve the model; however, such machine learning is not dependent on the functions of derivative information, such as the group of the intelligent optimization model is more suitable for solving the established model, the model for the solution of efficient. A new virtual network function service chain mapping algorithm PDQN-VNFSC was proposed by combining prediction algorithm and DQN (Deep Q-Network).

3. Deep Reinforcement Learning and DQN

Reinforcement learning (RL) model mainly describes the agent to interact with the environment repeatedly through the mechanism of trial and error and to learn the optimal strategy by maximizing the cumulative return. The model based on RL strategy consists of five key parts, including state S, action A, state transition probability P, return r, and strategy $\pi(s, a)$. In the process of interaction between agents and the environment, agents execute corresponding actions according to the strategy at different time points according to the observed state and system returns. After the action, the agent's state is transferred to the next state with the description of probability P, while the agent receives feedback from the environment in return r. Since the current state of the agent affects the next state and has nothing to do with the state before the current state, MDP can be used to describe the reinforcement learning model.

Through RL modeling, its core is to be able to get $\pi(s, a)$: $S \times A \longrightarrow [0, 1]$, that is, to get the mapping of the agent's state space and action space to probability. Generally, agents have huge state space and action space, which requires that the RL method can use limited learning experience to complete the acquisition and representation of effective knowledge in a large range of space. When the scale of operator network is large enough, the scale of system state space will make it difficult to solve the equations. More importantly, the state transition probability matrix of the SFC migration model cannot be obtained in advance, which makes it difficult to use both the classical strategy iteration method and the value iteration method. In the recent research work, deep neural network (DNN) has been successfully used to solve the reinforcement learning model and good results have been obtained.

3.1. SFC Deployment Based on Double DQN. When the network decision-making mechanism acts as an "agent" is in a certain state, it can choose a variety of actions, and the execution of different actions will make the agent enter the next different state. This paper introduces an action value function $Q^\pi(s, a)$ to estimate the value of each action. Thus, the action value function is represented as $Q^\pi(s, a) = E[r_{t+1} + \eta r_{t+2} + \eta^2 r_{t+3} + \cdots | (s, a)]$ and can be rewritten as

$$Q^\pi(s, a) = E_s[r + \eta Q^\pi(s', a') | (s, a)]. \tag{7}$$

In order to obtain the optimal strategy, we need to solve the optimal action value function:

$$Q^*(s, a) = E_{S'}[r + \eta \max_{a'} Q^*(s', a') | (s, a)]. \tag{8}$$

The value iteration algorithm is to update the Q value to make it converge to the optimal, and the idea of Q-learning is obtained completely according to the value iteration. However, value iteration needs to update all Q values each time. However, it is difficult to traverse the whole state space for the SFC deployment problem studied in this paper, so Q-learning only uses limited samples to update Q values:

$$Q(s, a) = (1 - \mu) Q(s, a) + \mu[r + \eta \max_{a'} Q^*(s', a') | (s, a)]. \tag{9}$$

Although the target Q value can be obtained according to the value iteration algorithm, it does not assign new Q value to the obtained Q value. Instead, it approaches the target in a progressive way, similar to gradient descent. In (9), the learning rate controlling the difference between the previous Q value and the new Q value can reduce the error. η represents the attrition rate, i.e., the degree to which future experience is important to the actions performed in the current state. Then, it converges to an optimal value of Q. In this paper, the value function $Q(s, a\colon \varphi)$ is introduced to represent the input of any state to get the output. The purpose is to transform the complex updating problem of the Q value into a function problem. Similar states correspond to similar actions to achieve the approximation of the value function and then continue to be expressed by $Q(s, a\colon \varphi) \approx Q*(s, a)$, where the parameters represent the weight of the neural network. By updating parameters, DQN makes the approximate Q function infinitely approximate to the optimal value and turns it into a function optimization problem. Due to the nonlinear characteristics

```
**Input:** Network Topology $G$, VNF-SC set $R$, VNF set VNF
**Output:** VNF-SC deployment strategy
(1) Initialize the neural network with random weight $\phi$;
(2) Initialize the action value function $Q$;
(3) Initialize the experience replay memory $ER$;
(4) for $episode = 1, 2, \ldots, G_{\max}$ do
(5)     Observation initial state$s^0$;
(6)     for $t = 1, 2, \ldots, ER$ do
(7)         A random $\theta$ is generated randomly;
(8)         if $\theta > \varepsilon$ then
(9)             Select a action $|a^t = \arg\max_{|a} Q(s_t, a: \phi_t)$;
(10)        else
(11)            Select a random action $a^t$ with probability $\varepsilon$;
(12)        end
(13)        Perform the action $a^t$ in the emulator and observe the return $r_{t+1}$ and the new state $s_{t+1}$;
(14)        Store the intermediate quantity $(s^t, a^t, r^t, s^{t+1})$ into the experiential pool memory $ER$;
(15)        Get a set of samples from the empirical pool memory $ER$;
(16)        Calculate the loss function $L|(\phi^t)$;
(17)        Calculate the gradient of the loss function with respect to $\phi^t$;
(18)        Update: $|\phi^t = \phi^t - \omega\nabla_{\phi^t} L(\phi^t)$, where $\omega$ is the learning rate;
(19)    end
(20) end
```

Algorithm 1: Framework of proposed algorithm.

of the function, this paper adopts the deep neural network as the approximate Q function; that is, the deep reinforcement learning method is adopted, and based on this, a method of SFC mapping based on double DQN is proposed, which effectively avoids the influence caused by overestimation. Dual DQN separates the selection and evaluation actions in the target Q value, allowing them to use different Q functions (network). One is used to generate the greedy strategy, and the other is used to generate the estimate value of the Q function, so the implementation needs two Q function networks. The Q function network of the original DQN is called the online network, and the latter is called the target network. The target used by the dual DQN algorithm can be expressed as

$$Y_t^{\mathrm{DQN}} = r_{t+1} + \nu Q\left(s_{t+1}, \arg\max_{a}{}' Q(s_{t+1}, a: \phi_t); \phi_t^-\right). \tag{10}$$

In the dual DQN, two different targets are calculated, respectively, from the current Q network and the target Q network. The current Q network is responsible for selecting actions, and the target Q network with delay ϕ_t^-, which is responsible for calculating the target Q value. In addition, the experience pool is used to solve the problem of correlation and nonstatic distribution. The experience pool stores the transfer samples (s^t, a^t, r^t, s^{t+1}) obtained from the interaction between each time-step agent and the environment to the playback memory unit. When training is needed, a part of the adjustment samples are randomly taken out for training.

The advantage of the algorithm based on dual DQN is that it can construct the loss function through Q learning and then solve the correlation and nonstatic distribution problems through experience replay. Meanwhile, it can use the target network. The network solves the stability problem. Algorithm 1 describes the pseudocode of the DQN-based SFC mapping algorithm.

4. Experimental Results and Analysis

4.1. Simulation parameters. In order to verify the effectiveness of the algorithm, simulation experiments are carried out in two widely used network topologies: the National Natural Science Foundation Network (NSFNet) with 14 nodes and 21 links and the US Backbone network with 27 nodes and 44 links. The number of frequency gaps on each link of the network is 1000, i.e., eight groups of different quantities (100, 200, . . ., 800). The initial occupancy frequency gap number of each group of virtual network function service chain requests is generated randomly in the interval [1, 10], and the change ratio of occupancy frequency gap number is generated randomly in the interval [0.5, 2]. It is assumed that there are altogether 10 virtual network functions, that is, the number of virtual network functions required by each virtual network function chain is generated within the interval [1, 10] and randomly divided into two kinds of virtual network functions that are dependent and independent of each other.

4.2. Experimental Results. In order to verify the effectiveness of the algorithm, this paper compares the two algorithms, respectively. The first algorithm (represented by LBA) proposed in [16] is compared. The second algorithm is a combination of LBA algorithm and the least-priority strategy proposed in [19] (represented by LF-LBA). Figure 1

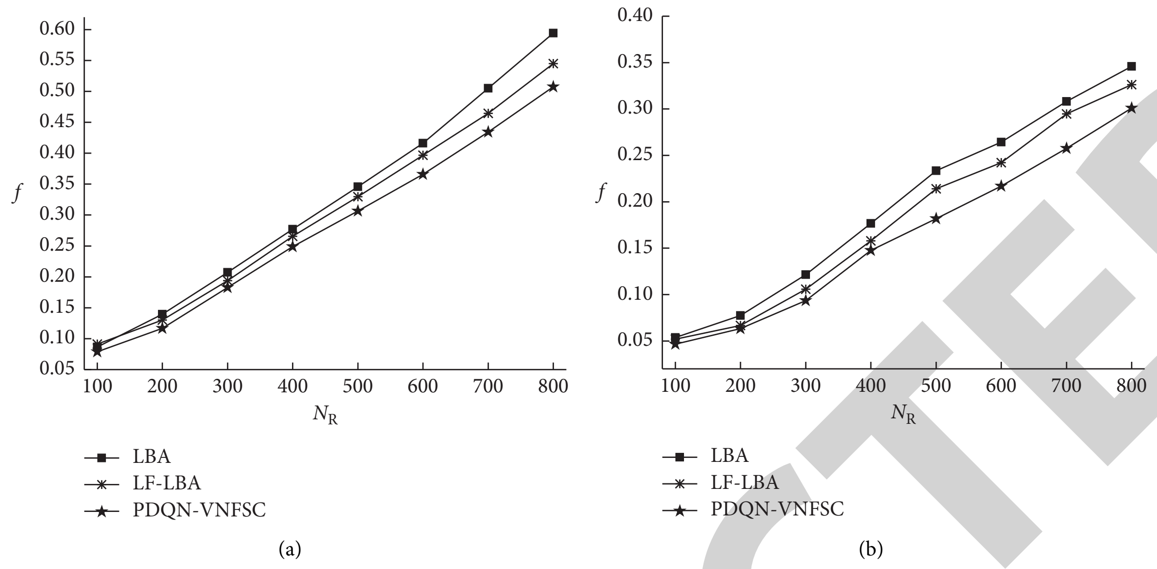

Figure 1: Experimental results when $N_{DC} = 1/2N_V$ and $\alpha = 1$: (a) NSFNET and (b) US backbone.

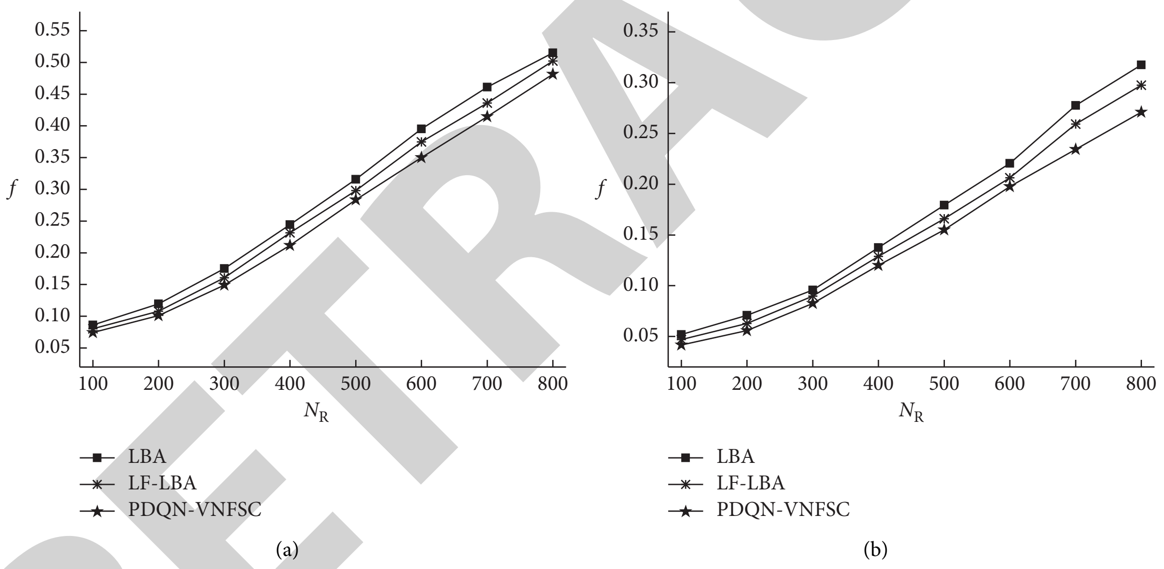

Figure 2: Experimental results when $N_{DC} = 1/2N_V$ and $\beta = 1$: (a) NSFNET and (b) US backbone.

shows the experimental results of NSFNET network and US Backbone network when $N_{DC} = 0.5N_V$ and $\alpha = 1$, respectively. Figure 2 shows the experimental results of NSFNET network and US Backbone network when $N_{DC} = 0.5N_V$ and $\beta = 1$, respectively. Figure 3 shows the experimental results of NSFNET network and US Backbone network when $N_{DC} = 0.5N_V$ and $\gamma = 1$, respectively. Figure 4 shows the experimental results of NSFNET network and US backbone network when $N_{DC} = 0.5N_V$ and $\alpha = \beta = \beta = 1/3$, respectively. Similarly, experimental results of NSFNET network and US Backbone network when $N_{DC} = 3/4N_V$ and $\alpha = 1$ are shown in Figure 5. Experimental results of NSFNET network and US backbone network when $N_{DC} = 3/4N_V$ and $\beta = 1$ are shown in Figure 6. Experimental results of NSFNET network and US backbone network when $N_{DC} = 3/4N_V$ and $\gamma = 1$ are shown in Figure 7. Experimental results of NSFNET network and US backbone network when $N_{DC} = 3/4N_V$ and $\alpha = \beta = \gamma = 1/3$ are shown in Figure 8.

4.3. Experimental Analysis. When $\alpha = 1$, the goal of optimization is to minimize the maximum frequency slots number occupying the frequency slots in the network. As can be seen from Figures 1 and 5, with the increase in the number of virtual network function service chains, the maximum frequency slots occupied in the network also increases gradually. Because the LBA algorithm does not consider the dependencies between virtual network functions and uses a fixed order to configure virtual network functions, it cannot well solve the configuration problem of

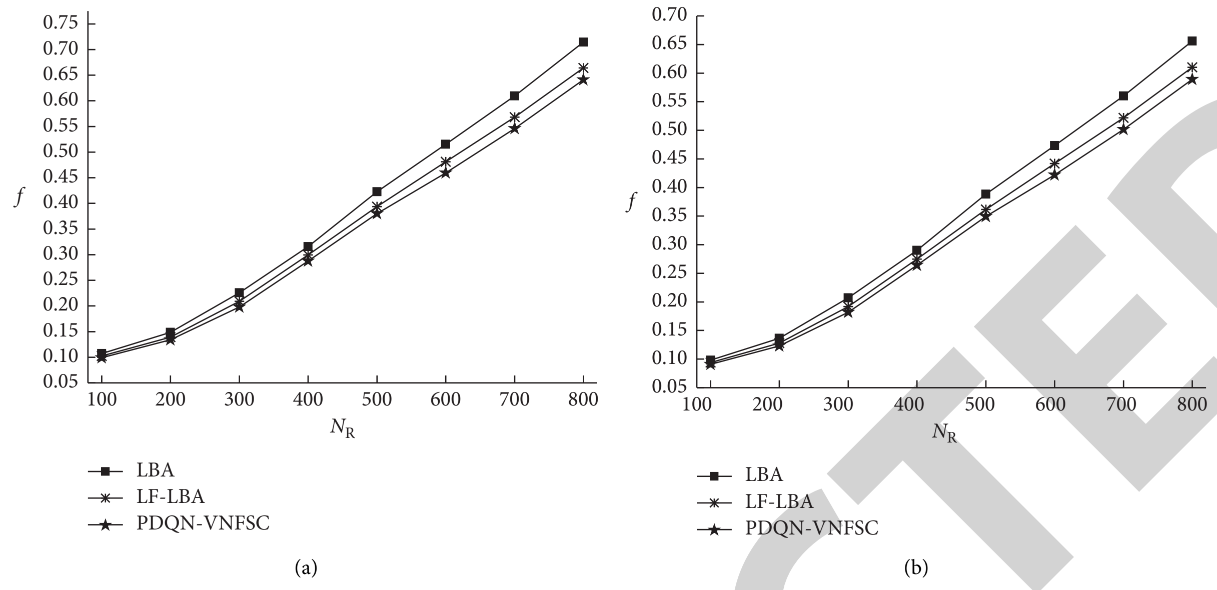

Figure 3: Experimental results when $N_{DC} = 1/2N_V$ and $\gamma = 1$: (a) NSFNET and (b) US backbone.

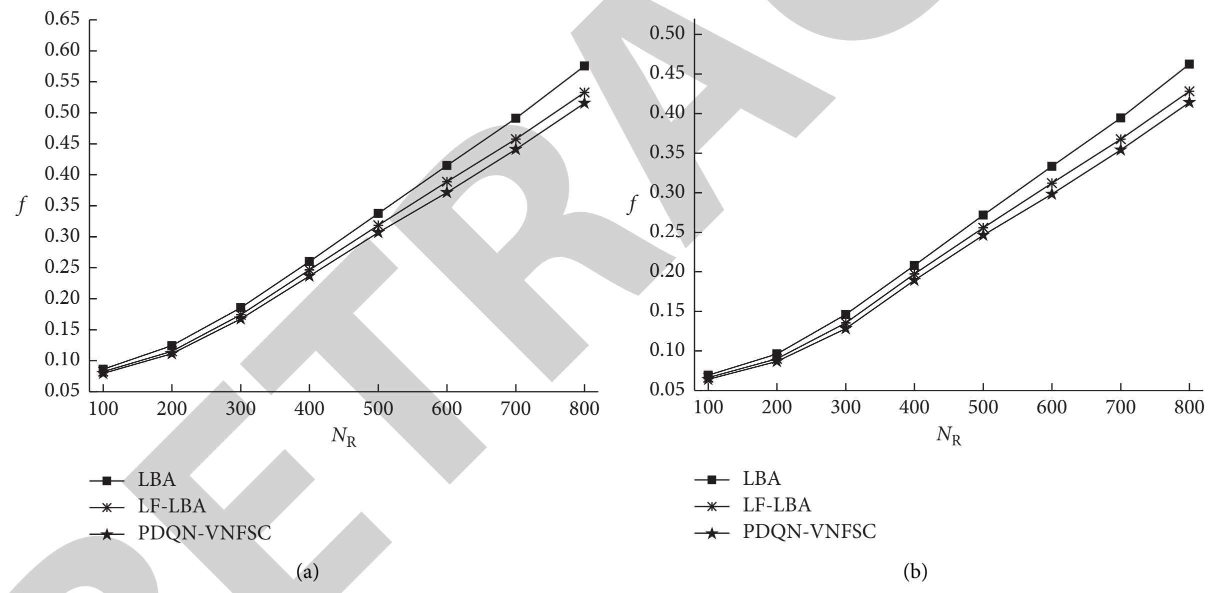

Figure 4: Experimental results when $N_{DC} = 1/2N_V$ and $\alpha = \beta = \gamma$ (a) NSFNET (b) US Backbone.

virtual network functions that consider the dependencies between virtual network service functions. The LF-LBA algorithm also fails to consider the dependencies among virtual network functions. Although the virtual network functions occupying a small frequency slots are given priority in configuration, the optimal scheme still cannot be obtained when the dependencies among virtual network functions are considered. However, the PDQN-VNFSC algorithm proposed in this paper takes into account the dependence between different virtual network functions and can search for the optimal configuration scheme through multiple iterations, so the PDQN-VNFSC proposed in this paper can obtain better results than LBA and LF-LBA. It can also be seen from the experimental results that the algorithm designed in this paper can get the maximum frequency slots occupied in the network which is smaller than the two contrast algorithms.When the number of virtual network functional service chains is 100, the PDQN-VNFSC algorithm proposed in this paper can get the maximum frequency slots occupied in the network 1.1%–2.3% smaller than the two contrast algorithms. When the number of virtual network functional service chains is 800, the PDQN-VNFSC algorithm proposed in this paper can obtain the maximum frequency slots occupied in the network, which is 5.1%–6.4% smaller than the two comparison algorithms. It can be seen from Figures 1 and 5; for the same network topology and the number of service chains with the same virtual network function, when the number of linked data centers in the network increases,

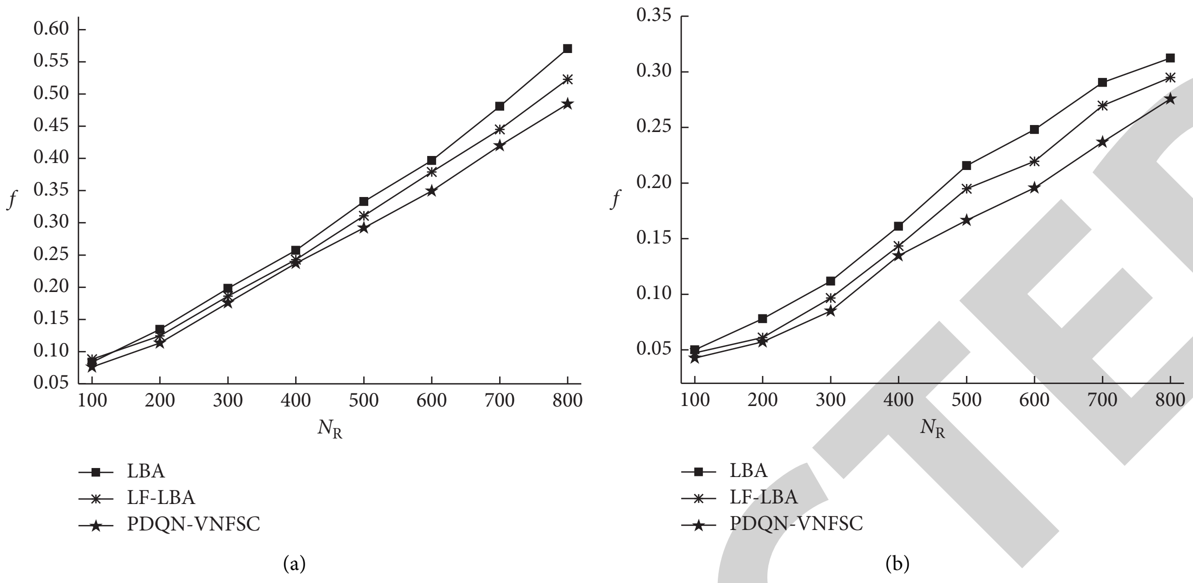

Figure 5: Experimental results when $N_{DC} = 1/2N_V$ and $\alpha = 1$: (a) NSFNET and (b) US backbone.

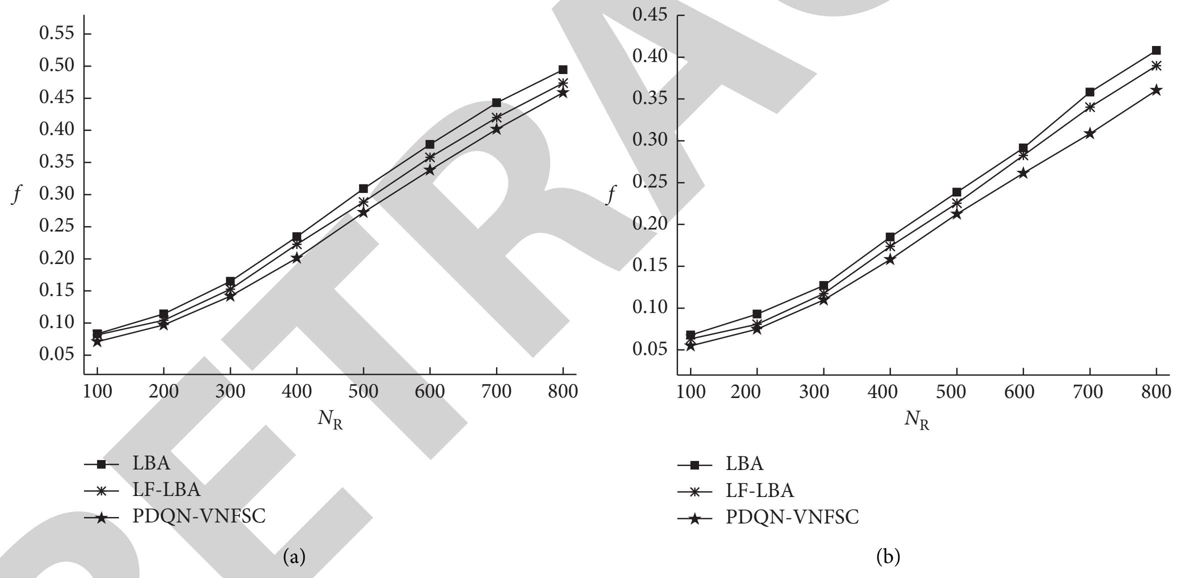

Figure 6: Experimental results when $N_{DC} = 1/2N_V$ and $\beta = 1$: (a) NSFNET and (b) US backbone.

the maximum frequency slots occupied in the network decreases. When the number of data centers is relatively small, these nodes in the network will become the key nodes, and more virtual network function service chains will pass through this node, which will also lead to more virtual network function service chains passing through the links connected with this node. Therefore, the maximum frequency slots occupied in the network will be larger. On the contrary, when the number of data centers is small, the virtual network function service chain will occupy different links more evenly, which will reduce the maximum frequency slots occupied in the network.

When $\beta = 1$, the objective of optimization is to minimize the number of frequency slots occupied in the network. As can be seen from Figures 2 and 6, with the increase in the number of virtual network function service chains, the number of frequency slots occupied in the network gradually increases. It can also be seen from the experimental results that the algorithm designed in this paper can get a smaller number of frequency slots occupied in the network than the two contrast algorithms. When the number of functional service chains in the virtual network is 100, the PDQN-VNFSC algorithm proposed in this paper can obtain the number of occupied frequency slots in the network which is 1.3%–2.4% smaller than the two comparison algorithms. When the number of functional service chains in the virtual network is 800, the PDQN-VNFSC algorithm proposed in this paper can obtain the number of occupied frequency slots

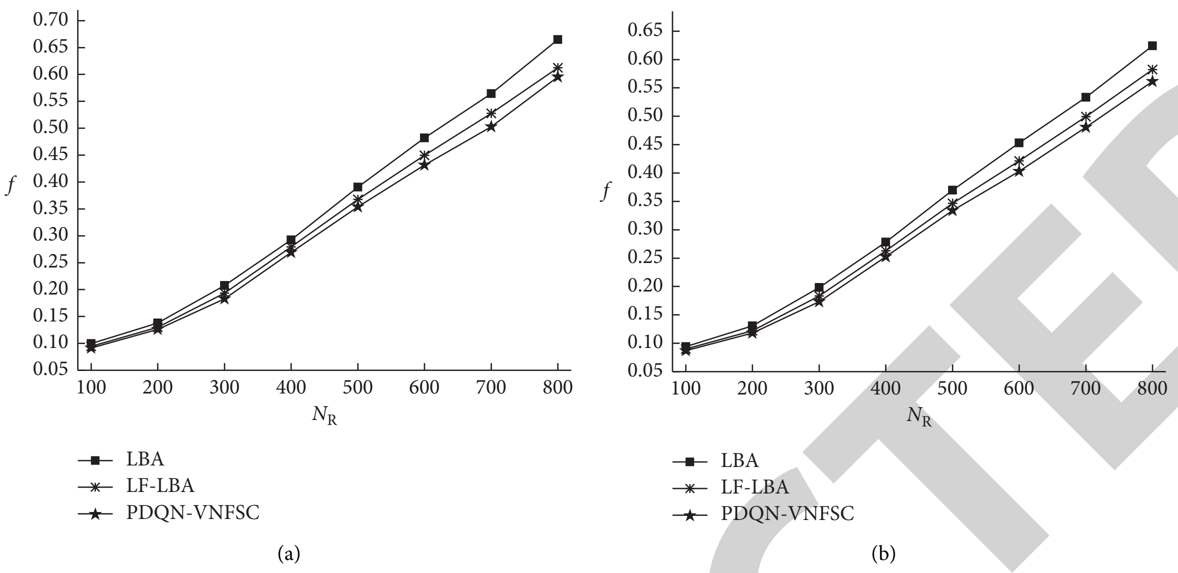

Figure 7: Experimental results when $N_{DC} = 1/2N_V$ and $\gamma = 1$: (a) NSFNET and (b) US backbone.

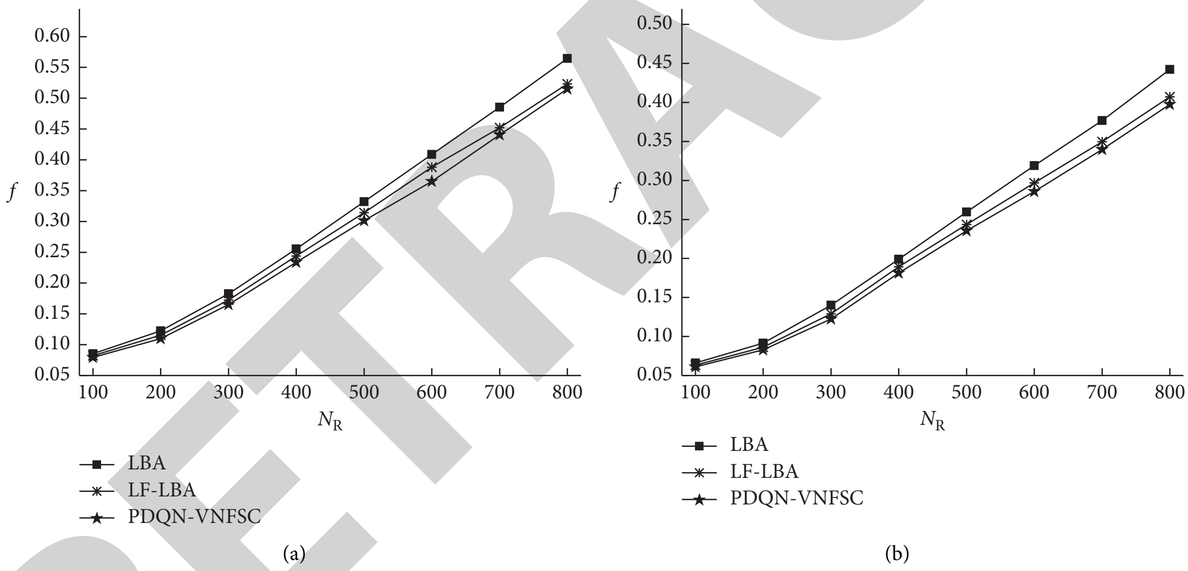

Figure 8: Experimental results when $N_{DC} = 1/2N_V$ and $\alpha = \beta = \gamma$: (a) NSFNET and (b) US backbone.

in the network which is 4.2%–5.6% smaller than the two comparison algorithms.

When $\gamma = 1$, the goal of optimization is to minimize the number of virtual network functions configured on the data center in the network. As can be seen from Figures 3 and 7, as the number of virtual network function service chains increases, the number of virtual network functions configured on the data center in the network also gradually increases. The experimental results also show that the algorithm designed in this paper can get fewer virtual network functions configured on the data center in the network than the two contrast algorithms. When the number of virtual network function service chains is 100, the PDQN-VNFSC algorithm proposed in this paper can obtain that the number of virtual network functions configured on the data center in the network is 1.0%–1.9% smaller than the two comparison algorithms. When the number of virtual network function service chains is 800, the PDQN-VNFSC algorithm proposed in this paper can get that the number of virtual network functions configured on the data center in the network is 3.4%–4.6% less than the two comparison algorithms.

When $\alpha = \beta = \gamma = 1/3$, the goal of optimization is to minimize the maximum frequency slots number occupying frequency slots in the network, the number occupying frequency slots in the network, and the number of virtual network functions configured on the data center in the network, and the three have the same weight. As can be seen from Figures 4 and 8, the three target values after weighted summation gradually increase with the increase of the number of virtual network function service chains. It can also be seen from the experimental results that the algorithm designed in this paper can obtain the three target values after the weighted sum less than the two contrast algorithms. When the number of virtual network functional service chains is 100, the PDQN-VNFSC algorithm proposed in this paper can get the three target values after the weighted sum, which are 1.7%–4.2% smaller than the two comparison algorithms. When the number of virtual network functional service chains is 800, the PDQN-VNFSC algorithm proposed in this paper can get the three target values after the weighted sum, which is 3.4%–4.8% smaller than the two comparison algorithms.

5. Conclusion

In order to solve the routing problem of virtual network function service chain and the configuration problem of virtual network function in elastic optical network, a global constraint optimization model is established, which aims at minimizing the maximum frequency slots occupied in the network, the number of frequency slots occupied in the network, and the number of virtual network functions configured.The model divides the virtual network function into two types: the virtual network function with dependence and the virtual network function without interdependence. In order to solve the model efficiently, the weighted summation method was used to transform the three-objective optimization problem into a single-objective constrained optimization problem, and a deep reinforcement learning-based algorithm was designed. However, this paper uses the weighted method to transform the three-objective optimization problem into a single-objective constrained optimization problem. In the following research, the multiobjective optimization method will be used to solve the established model, so as to provide more decision schemes for decision makers.

Data Availability

All the data used to support the findings of the study are available within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by National Natural Science Foundation of China (no. 31872704), Science and Technology Department of Henan Province (nos. 202102210161 and 212102210392), and Nanhu Scholars Program for Young Scholars of XYNU.

References

[1] M. C. Luizelli, W. C. Cordeiro, L. S. Buriol, and L. P. Gaspary, “A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining,” *Computer Communications*, vol. 102, no. 1, pp. 67–77, 2016.

[2] X. Xue, “A compact firefly algorithm for matching biomedical ontologies,” *Knowledge and Information Systems*, vol. 62, no. 11, Article ID 2855C2871, 2020.

[3] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, “Network function virtualization: state-of-the-art and research challenges,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2017.

[4] Y. Li and M. Chen, “Software-defined network function virtualization: a survey,” *IEEE Access*, vol. 3, pp. 2542–2553, 2017.

[5] I. F. Akyildiz, S.-C. Lin, and P. Wang, “Wireless software-defined networks (w-sdns) and network function virtualization (nfv) for 5g cellular systems: an overview and qualitative evaluation,” *Computer Networks*, vol. 93, no. 24, pp. 66–79, 2015.

[6] M. Otokura, K. Leibnitz, T. Shimokawa, and M. Murata, “Evolutionary core-periphery structure and its application to network function virtualization,” *Nonlinear Theory and Its Applications, IEICE*, vol. 7, no. 2, pp. 202–216, 2016.

[7] H. Xuan, X. Zhao, L. You, Z. Liu, and Y. Li, “Multiobjective model and improved artificial raindrop algorithm for virtual network mapping,” *Mobile Information Systems*, vol. 2021, Article ID 5542670, 10 pages, 2021.

[8] M. Zeng, W. Fang, and Z. Zhu, “Orchestrating tree-type vnf forwarding graphs in inter-dc elastic optical networks,” *Journal of Lightwave Technology*, vol. 34, no. 14, pp. 3330–3341, 2016.

[9] A. Bradai, M. H. Rehmani, I. Haque, M. Nogueira, and S. H. R. Bukhari, “Software-defined networking (sdn) and network function virtualization (nfv) for a hyperconnected world: challenges, applications, and major advancements,” *Journal of Network and Systems Management*, vol. 28, no. 3, pp. 433–435, 2020.

[10] L. Qu, C. Assi, and K. Shaban, “Delay-aware scheduling and resource optimization with network function virtualization,” *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3746–3758, 2016.

[11] W. Miao, G. Min, Y. Wu et al., “Stochastic performance analysis of network function virtualization in future internet,” *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 3, pp. 613–626, 2019.

[12] X. Cheng, Y. Wu, G. Min, and A. Y. Zomaya, “Network function virtualization in dynamic networks: a stochastic perspective,” *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2218–2232, 2018.

[13] H. Xuan, X. Zhao, Z. Liu, J. Fan, and Y. Li, “Energy efficiency opposition-based learning and brain storm optimization for vnf-sc deployment in iot,” *Wireless Communications and Mobile Computing*, vol. 2021, no. 9, 9 pages, Article ID 6651112, 2021.

[14] L. Tachun, Z. Zhili, M. Tornatore, B. Mukherjee, Demand-aware network function placement,” *Journal of Lightwave Technology*, vol. 34, no. 11, pp. 2590–2600, 2016.

[15] M. Mechtri, C. Ghribi, and D. Zeghlache, “A scalable algorithm for the placement of service function chains,” *IEEE transactions on network and service management*, vol. 13, no. 3, pp. 533–546, 2016.

[16] M. Karimzadeh-Farshbafan and V. Shah-Mansouri, D. Niyato, A dynamic reliability-aware service placement for network function virtualization (nfv),” *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 318–333, 2019.

[17] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: joint sampling and matching approach," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 172–185, 1939.

[18] P. T. A. Quang, Y. Hadjadj-Aoul, and A. Outtagarts, "A deep reinforcement learning approach for vnf forwarding graph embedding," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1318–1331, 2019.

[19] J. Pei, P. Hong, M. Pan, J. Liu, and J. Zhou, "Optimal vnf placement via deep reinforcement learning in sdn/nfv-enabled networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 263–278, 2020.

[20] R. Solozabal, J. Ceberio, A. Sanchoyerto, L. Zabala, and F. Liberal, "Virtual network function placement optimization with deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 292–303, 2019.

[21] L. Baojia, L. Wei, Z. Zuqing, Deep-nfvorch: leveraging deep reinforcement learning to achieve adaptive vnf service chaining in dci-eons," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 12, no. 1, pp. A18–A27, 2019.