Journal of Network and Systems Management (2022) 30: 58
https://doi.org/10.1007/s10922-022-09673-5

Multi-objective Optimization Service Function Chain Placement Algorithm Based on Reinforcement Learning

Hongtai Liu[1] · Shengduo Ding[1] · Shunyi Wang[1] · Gang Zhao[1] · Chao Wang[2]

Received: 31 August 2021 / Accepted: 3 July 2022 / Published online: 16 July 2022

Abstract

Network function virtualization (NFV) makes the realization of specific network functions no longer depend on inherent hardware by executing virtual network functions (VNFs), but realizes network functions in a more flexible programming manner, thereby reducing the pressure of resource allocation on the underlying network. Service function chain (SFC) is composed of a set of fixed order VNFs. These VNFs need to be deployed on appropriate physical nodes to meet user function requirements, i.e., the placement of SFC. Traditional solutions mostly use mathematical models or heuristic methods, which are not applicable in the context of large-scale networks. Secondly, the existing methods do not integrate intelligent learning algorithms into the service function chain placement (SFCP) problem, which limits the possibility of obtaining better solutions. This paper presents a multi-objective optimization service function chain placement (MOO-SFCP) algorithm based on reinforcement learning (RL). The goal of the algorithm is to optimize the resource allocation mode, including several performance indexes such as underlying resource consumption revenue, revenue cost ratio, VNF acceptance rate and network latency. We model the SFCP as a Markov decision process (MDP), and use a two-layer policy network as an intelligent agent. In the training stage of RL, the agent comprehensively considers the optimization objectives and formulates the optimal physical node mapping strategy for VNF requests. In the test phase, the whole SFCP is completed according to the trained node mapping strategy. Simulation results show that the algorithm proposed in this paper has excellent performance in the aspects of underlying resource allocation revenue, VNF acceptance rate and so on. In addition, we prove that the algorithm has good flexibility by changing the delay constraint.

Keywords Network function virtualization · Virtual network function · Service function chain placement · Reinforcement learning · Markov decision process

✉ Chao Wang
wangch_upc@qq.com

Extended author information available on the last page of the article

1 Introduction

Internet architecture has effectively served the development of science and technology society in the past decades [1]. With the advent of the intelligent era, the traditional Internet architecture has gradually become rigid. A typical feature of the modern network environment is that user function requests arrive on a scale of hundreds of millions. Internet service providers (ISPs) need to allocate specific network resources to massive user requests in a short period of time. The actual result is that the ISPs' network resource scheduling method will always lead to higher capital expenditure and operating expenditure, and the resource allocation method is extremely inflexible [2, 3]. What is exciting is that network virtualization (NV) technology came into being, which redefines the allocation of physical network resources [4, 5]. Based on network function virtualization (NFV), virtual network function (VNF) can replace manufacturer's dedicated hardware. NFV performs the same functions as proprietary hardware through software programming, thus effectively improving the flexibility of the physical network [6–8].

Service function chain placement (SFCP) is a typical network service paradigm, which is composed of a set of fixed and orderly VNFs. Network traffic is required to flow through the VNFs in a predefined order [9]. Therefore, the essence of SFCP is the allocation process of physical network resources. It is necessary to explore an efficient SFCP method to improve the efficiency of network resource orchestration. Its purpose is to improve the revenue of resource allocation and reduce the cost of resource consumption on the basis of receiving as many VNF requests as possible. SFCP is a NP hard problem [10]. Traditional solutions include establishing mathematical model or using heuristic methods. However, with the continuous expansion of the scale of underlying network, the applicability and time efficiency of the solution based on mathematical model are getting lower and lower. In addition, heuristic methods lack strict theoretical proof, and the results often fall into local optimal solutions [11].

In recent years, machine learning (ML) algorithm has been paid enough attention, and it has made breakthrough progress in finance, medicine, automation and Internet [12]. As a typical representative of ML, reinforcement learning (RL) has won the favor of researchers with its excellent decision-making ability, and plays a crucial role in decision-making in high-dimensional space [13]. RL consists of three main elements, action a, state s and reward r. The performer of the action is called an agent. It continuously carries out interactive learning (training process) with the environment, and finally obtains a mapping from the environment to the action. In this process, the state of environment has changed. The goal of learning is to maximize the cumulative reward. It can try to master the characteristics of network resources with the excellent learning ability of RL, so as to effectively manage network resources and provide the good solution for SFCP.

In general, the following key challenges are faced in the implementation of SFCP under virtual network architecture [14]. Firstly, the Internet structure is composed of a large number of physical and hardware devices. Heterogeneous network resources (CPU, memory, bandwidth, etc.) are stored in various distributed storage devices in a decentralized state. In addition, the network resource state will change constantly due to the user function request, and the network topology will even be in dynamic change. Second, how to select the best target physical node for VNF requests, while taking into account performance such as revenue, cost, and load balancing. Therefore, the VNF placement problem under virtual network architecture has a broad research space.

Network resource allocation is the core business and key of network function execution. Many network performance problems and defects can be solved by means of network resource scheduling. However, the multi-objective optimization of network resource allocation, i.e., the SFCP in virtual network architecture, has not been effectively solved. The algorithm of multi-objective optimization service function chain placement (MOO-SFCP) based on RL is divided into training stage and running stage. Firstly, we model the SFCP as a Markov decision process (MDP). Afterwards, based on RL theory, we use a two-layer policy network as an intelligent agent and make it participate in training, with the purpose of making the optimal decision for SFCP. The main function of RL is to improve the efficiency of the algorithm.

The main contributions of this paper are as follows.

1. We analyze the problem of SFCP in virtual network architecture, focusing on optimizing multiple goals such as network resource distribution revenue, resource consumption costs, VNF request acceptance rate and load balancing.
2. We model the SFCP as a MDP and use RL method to solve this problem. In this paper, a self built two-layer policy network is used as the agent to participate in the training and operation process, in order to derive the optimal strategy of SFCP.
3. Through comparative experiments, we prove the excellent performance of the MOO-SFCP algorithm based on RL. In terms of improving resource revenue and request acceptance rate, and reducing resource consumption costs, the algorithm proposed in this paper has advantages over other virtual network algorithms. In addition, the flexibility of the algorithm is verified by changing the latency constraint.

The structure of the paper is as follows. Section 2 introduces the research progress of VNF and SFCP. Section 3 describes the related concepts and gives the system model. Section 4 formulates the problems related to the SFCP. Section 5 introduces the implementation process of MOO-SFCP algorithm based on RL. In Sect. 6, the experimental simulation is carried out and the results are analyzed. Finally, we summarize the whole paper in Sect. 7.

2 Related Work

As a promising future network architecture, virtual network has attracted the attention of the industry, and relevant personnel have carried out extensive research on VNF request and SFCP. On the whole, there are three solutions for VNF request and SFCP. The first is to establish a mathematical model for the problem and provide a solution from the perspective of mathematical optimization. The second is the heuristic method, and the third is the solution based on ML algorithm. The following analyzes the research status of SFCP from these three aspects.

The existing researches usually define the problem as integer linear programming (ILP) model [15–17], mixed integer linear programming (MILP) model [18–21] or binary integer programming (BIP) model [22–24].

Li et al. [15] studied the placement of VNFs in edge computing networks. Because the edge computing network is a hierarchical network, there are many restrictions on placing VNFs in it. The authors modeled the problem as an ILP model and took minimizing energy consumption as the goal. When the virtual network had fewer functions, the optimization solver was used to obtain a more ideal result. In addition, they also transferred the VNF placement problem to the cloud data center for research [16]. With the goal of minimizing the number of physical machines, they proposed a two-stage heuristic method using greedy algorithm and adjustment algorithm. The simulation results showed that the algorithm can effectively improve the utilization of network resources. Qi et al. [17] solved the problem of non-scalability of VNFs. By limiting a small searchable range, the search space of VNFs was effectively reduced, and the efficiency of SFCP was improved.

Tang et al. [18] introduced a specific method to implement dynamic VNF placement. Specifically, they put forward a traffic prediction method by analyzing the business characteristics of network operators, and then realized the expansion of dynamic VNFs through VNF positioning algorithms. Hawilo et al. [19] modeled VNF placement as a MILP model, and proposed a central scheduling algorithm to optimize the end-to-end latency of SFC. In addition, the authors also verified the reliability and service quality of the algorithm by adjusting the number of VNFs, and analyzed the complexity of the algorithm in detail. Reference [20] studied the VNF mapping problem in the space-air-ground integrated network, and applied it to Internet of vehicles scenarios. The authors modeled the real-time migration of VNFs as MILP, and then proposed two heuristic algorithms based on tabu search. The results showed that the proposed scheme was close to the optimal solution.

Pei et al. [22–24] unified the modeling of VNF placement as BIP model. From many perspectives, the dynamic release of VNFs, the consumption of SFC request resources, and the minimization of end-to-end latency of each VNF were studied. The SFCP scheme based on mathematical model is convenient and effective when dealing with small-scale problems, and has high accuracy and efficiency. But it cannot be ignored that this method will increase the complexity and reduce the efficiency when solving large-scale network problems.

Heuristic method is an effective solution to deal with the SFCP in large-scale network. Bari et al. [25] proposed a heuristic algorithm based on dynamic programming to orchestrate VNF instances. After that, the authors conducted a simulation in a real network environment, and the results proved that the network operating cost was greatly reduced. Based on feature decomposition method, Mechtri et al. [26] proposed a custom heuristic greedy algorithm to find the best placement of VNFs. In addition, the algorithm also effectively solved the connection problem of the SFC. Reference [27] proposed a method based on Markov approximation to solve the problem of flow perception and cost minimization of

VNFs. In order to optimize the time performance, the authors proposed to combine the matching theory with the Markov approximation method. The results proved that the cost of resource allocation can be reduced. Although the heuristic method effectively lifted the time limit for the SFCP, its feasibility still needs to be rigorously proved by theory.

ML, as a new learning paradigm, is more and more used in real life. Scholars have also carried out relevant research on the VNF request and SFCP based on ML. Santos et al. [28] investigated the impact of different SFCP policies on traffic and latency in highly distributed scenarios, and discussed several representative SFCP scenarios and techniques. Reference [29] considered the problem of VNF relocation. Taking into account the user location and the resource status of the currently placed nodes, the authors proposed a dynamic VNF migration method, which effectively reduced the blocking rate of SFCP. Pei et al. [30] proposed a SFCP algorithm based on dual-depth Q-network. Based on DRL, the algorithm model can determine the best scheme from the huge search space, and then placed or released VNF instances based on threshold rules. The evaluation showed that the algorithm has excellent performance in throughput, delay and load balancing. The authors of reference [31] optimized the search space of VNFs based on RL and explored the application effects of new learning techniques in VNF forwarding graphs. The authors found through practice that general learning methods cannot effectively search large-scale spaces. So from the perspective of satisfying QoS, they designed a VNF forwarding graph allocation scheme based on RL. Sun et al. [32] considered the resource optimization and service quality issues of VNF placement. The authors tried to combine deep reinforcement learning (DRL) with graph neural network (GNN), and explored a new VNF placement scheme.

In order to solve the problems of high deployment latency of VNF service chain and difficulty in network management, Li et al. [33] proposed a resource pre-deployment management framework. The framework was essentially a discrete time system. In pre-deployment phase, the deep learning (DL) model was used to predict VNF service requests, and the SFC was deployed in execution phase. Troia et al. [34] studied the resource allocation of dynamic SFC based on RL in core optical networks. RL agent can learn resource allocation strategies from dynamically changing networks, and independently construct self-learning systems that can solve high-dimensional complex problems. RL agent made the best resource allocation decision based on the network status and historical traffic conditions. The authors of [35] studied the SFCP in mobile edge computing systems with the goal of minimizing latency. The authors first modeled SFCP as a flexible job-shop scheduling problem, and then proposed an adaptive scheduling algorithm based on DRL with the goal of minimizing the system latency, which realized the efficient scheduling of SFC. Xiao et al. [36] proposed an adaptive online SFC deployment algorithm based on DRL to deal with the changes of network and service requests. The authors used MDP model to capture the real-time changing network state, and then used serialization backtracking method to deal with large discrete space. Finally, it effectively reduced the cost of network operators and improved the system throughput. We also use MDP to model SFC, but we use a new method to characterize the underlying network state, and we are different from this work in terms of optimization objectives.

The obvious disadvantage of SFCP scheme based on the mathematical model is that it is not suitable for large-scale networks. Although the heuristic-based solution can effectively make up for this shortcoming, the results of this method often fall into the local optimal solution, and it cannot guarantee that the solution obtained is the optimal strategy for the placement of the service function chain. The solution based on ML has obvious advantages, but the existing research ignores the multi-objective optimization of network resource allocation. Moreover, most of them ignore the dynamic change of the underlying network and user requests, and do not show the specific situation of network resources. The SFCP algorithm based on RL is proposed in this paper, which focuses on the multi-objective optimization of resource allocation, and the RL agent can extract time-varying network information as training data effectively. Therefore, the work done in this paper is obviously different from the existing work.

3 Problem Description and Network Model

3.1 Problems Related to SFCP

Generally speaking, in a network scenario, a data stream usually flows through multiple network devices, such as firewalls, intrusion detection/defense systems (ID/PS), load balancers, etc., and finally reaches the destination. The beginning to the end of the data flow can be seen as a complete SFC. SFC is a popular network service paradigm, which consists of a set of VNF sequences, and the data flow needs to flow through specific network functions in a prescribed order. Therefore, under the condition of limited physical resources, how to select appropriate physical network nodes for VNFs for mapping, and how to construct a reasonable SFC path for data traffic has become a key issue. A typical example of SFC is shown in Fig. 1. Many studies have shown that SFCP is a NP-hard problem. There are usually two ways to solve the problem of SFCP, one is horizontal solution, the other is vertical solution. The former refers to the number of VNF requests that can be operated in the virtual network, but the resource amount of each function request is constant. The latter refers to the number of resources requested by each VNF that can be operated, but the number of VNF participating in the service function chain remains unchanged. This paper mainly adopts the first research idea, that is to study the resource optimization problem of SFCP under the variable number of VNF requests.

3.2 Reinforcement Learning and Markov Decision

RL treats the learning process as a trial and error process. The agent exerts an action on the environment. When the environment is stimulated by the action, the state will change, and a reward signal will be generated to feed back to the agent. After that, the agent will choose the next action according to the size or positive or negative of the reward signal, in order to maximize the cumulative reward signal. The essence of RL is the mapping from environmental state to action. Thanks

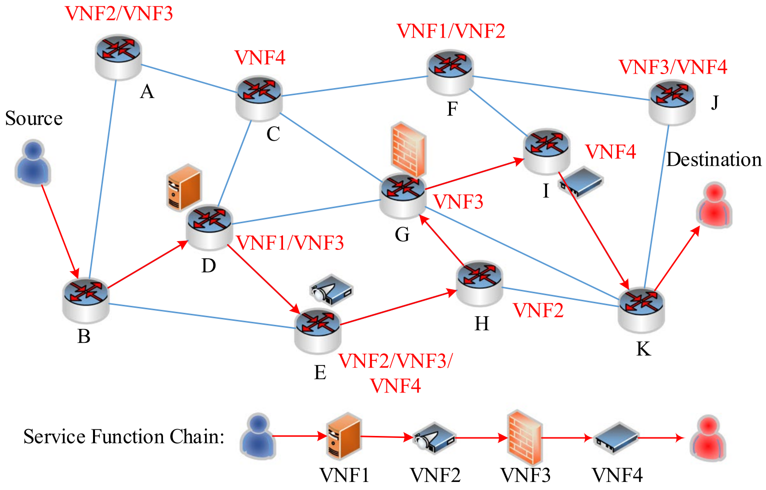


Fig. 1 Typical service function chain placement example

to the agent's autonomous learning ability, RL can achieve good decision-making effect, so it is widely used in decision-making problems in high-dimensional space.

Markov decision is a typical RL algorithm model, which can maximize the agent's reward through dynamic programming or random sampling. MDP can be expressed as a quadruple $M = (s_t, a_t, p_{s_t a_t}, r_t)$, where $s \in S$ represents the state set, and s_t represents the state of the environment at time t. $a \in A$ represents a set of actions, and a_t represents the actions taken by the agent at time t. $p_{s_t a_t}$ indicates that in state s_t, by applying action a_t to the environment, the environment state transitions to other probability distributions. For example, $p(s_{t+1|s_t,a_t})$ indicates the probability of transition to state s_{t+1} when action a_t is applied to the environment in state s_t. r_t represents the reward that the agent will get when the environment applies action a_t in state s_t, namely $s_t \times a_t \mapsto r_t$.

RL has the characteristics of delayed reward, so for any previous state s_t and action a_t, the immediate reward function $r_t(s_t, a_t)$ may not be able to prove the advantages and disadvantages of the current strategy, so a value function needs to be defined to indicate the long-term impact of the strategy π_t in the current state. In the SFCP problem, the strategy indicates the specific SFCP method. We list the commonly used RL value functions in Table 1, where $V^{\pi}(s)$ represents the value function of the state s obtained by adopting the strategy π, and r_t represents the immediate reward at time t.

Given strategy π_t, state s_t and action a_t, turning to state s_{t+1} with probability $p(s_{t+1}|s_t, a_t)$ at the next moment, then the state value function can be defined as,

$$V^{\pi_t}(s_t) = \sum_{s_{t+1} \in S} p(s_{t+1}|s_t, a_t)[r(s_{t+1}|s_t, a_t) + \gamma V^{\pi_t}(s_{t+1})]. \tag{1}$$

Table 1 Reinforcement learning value function

Function	Description
$V^{\pi}(s) = E_{\pi}[\sum_{t=0}^{T} r_t \| s_0 = s]$	When the strategy π is adopted, the sum of the expected immediate reward in the future time T
$V^{\pi}(s) = \lim_{T\to\infty} E_{\pi}[\frac{1}{T}\sum_{t=0}^{T} r_t \| s_0 = s]$	The average reward expected when the strategy π is used
$V^{\pi}(s) = E_{\pi}[\sum_{t=0}^{\infty} \gamma^t r_t \| s_0 = s]$	The most common form of value function. $\gamma \in [0,1]$ is called the conversion factor, which indicates the importance of future rewards relative to the current rewards

The action value function is defined as follows,

$$\begin{aligned} Q^{\pi_t}(s_t, a_t) &= E[\sum_{t=1}^{\infty} \gamma^t r_t | s_0 = s_t, a_0 = a_t] \\ &= \sum_{s_{t+1}\in S} p(s_{t+1}|s_t, a_t)[r(s_{t+1}|s_t, a_t) + \gamma V^{\pi_t}(s_{t+1})]. \end{aligned} \tag{2}$$

The main difference between the two value functions is that the action of state value function is determined by strategy π_t and state s_t, while the action of action value function is artificially defined. Therefore, our ultimate goal is to find the strategy π^* that can maximize the value function in the initial state s_0.

$$\pi^* = argmax_{\pi_t} V^{\pi_t}(s_t). \tag{3}$$

3.3 Physical Network Model

The physical network is represented by an undirected graph model $G^p = \{N^p, E^p, A^p\}$. The element N^p represents the node collection of the physical network, E^p represents the link collection of the physical network, and A^p represents the resource attribute collection of the physical network. Specifically, we take CPU, bandwidth and delay as the measurement of network resource attributes, namely $A^p = \{C, B, D\}$. The above three attributes are important resource attributes in physical network. First, we need to measure the revenue and cost of resource allocation in terms of CPU and bandwidth. Second, we need to use delay to test the flexibility of the algorithm. We use n^p to represent a specific physical node, and C_{n^p} represents the CPU resource capacity of node n^p. We use $e^p_{(x,y)}$ to represent the physical link between node n^p_x and node n^p_y, and $B^p_{(x,y)}$ represents the bandwidth resource capacity of the physical link between node n^p_x and node n^p_y. In order to simplify the network structure, we integrate the physical link attributes into the adjacent physical nodes, and use D_{n^p} to represent the delay attribute value of node n^p.

Table 2 Network symbols

Symbol	Description	Symbol	Description
G^P	physical network	N^P	Collection of physical nodes
E^P	collection of physical links	A^P	Collection of physical network attributes
C	CPU resources of physical nodes	B	Bandwidth resources of physical links
D	delay attributes of physical nodes	G^f	Service function chain
N^f	virtual network function nodes	E^f	User request links between VNF nodes
R^f	resource requirements attribute of SFC	RC	CPU resource requirements of VNF
RB	bandwidth resource requirements of user request links	RD	Delay requirements of VNF

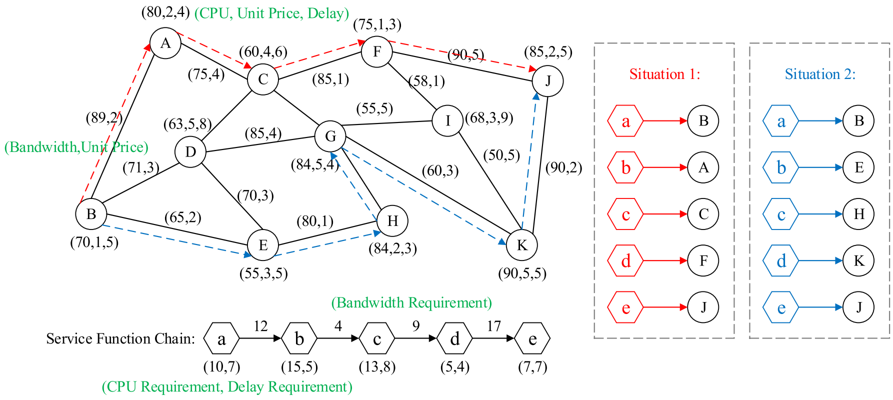


Fig. 2 Service function chain placement model and possible placement

3.4 Service Request Model

The resource requirements that users send to the physical network are called user requests. In the virtual network environment, this request is called a VNF request. Therefore, the SFC can be seen as a series of user requests. The SFC is represented by a directed acyclic graph $G^f = \{N^f, E^f, R^f\}$. The element N^f represents the VNF collection of the SFC, E^f represents the link collection between the VNFs, and R^f represents the resource requirement attribute collection of the SFC. In particular, $R^f = \{RC, RB, RD\}$. We use n^f to represent a specific VNF, and RC_{n^f} to represent the CPU resource demand of the VNF n^f. We use $e^f_{(i,j)}$ to represent the link between the VNF n^f_i and the VNF n^f_j, and $RB^f_{(i,j)}$ represents the bandwidth resource requirement of the link between the VNF n^f_i and the VNF n^f_j. RD_{n^f} represents the delay requirement of the VNF n^f_i.

We summarize all relevant network symbols in Table 2.

We abstract the typical SFCP situation as the network structure shown in Fig. 2. The circle represents the physical node and the hexagon represents the VNF request node.

The purpose of SFCP is to map the VNF request nodes to the physical nodes that meet the requirements of resource attributes, and the link resource attributes between nodes also meet the requirements. The number next to the physical node represents CPU resource, CPU resource unit price and delay attribute in turn, and the number next to the link represents bandwidth resource and bandwidth resource unit price in turn. The number next to the VNF represents CPU resource demand and delay demand in turn, and the number on the SFC link represents bandwidth demand. Figure 2 shows the possible placement of two SFCs. Because the two adjacent VNFs may not be mapped to the two adjacent physical nodes, the SFC link segmentation may occur, which is the second case shown in Fig. 2. This situation will lead to the consumption of more link resources.

4 Problem Formulation

4.1 Constraint Condition

The mapping of SFC to the underlying physical network will occupy the corresponding part of network resources. In addition, SFC has a certain life cycle. Only when SFC leaves, the occupied network resources will be released. Therefore, the SFC cannot be mapped to the physical network indefinitely, and it needs to follow specific constraints. At a certain time t, the available CPU resource capacity of physical node n^p and the available bandwidth resource capacity of physical link $e^p_{(x,y)}$ are defined as follows,

$$R\_C_{n^p} = C_{n^p} - \sum_{n^f_i \uparrow n^p}^{m} RC_{n^f_i}, n^f_i \in N^f, \tag{4}$$

$$R\_B_{e^p_{(x,y)}} = B_{e^p_{(x,y)}} - \sum_{e^f_{(i,j)} \uparrow e^p_{(x,y)}} RB^f_{(i,j)}, e^f_{(i,j)} \in E^f. \tag{5}$$

In Eq. (4), $n^f_i \uparrow n^p$ represents the VNF n^f_i mapped to the physical node n^p, and m represents the total number of virtual nodes mapped to the n^p. In Eq. (5), $e^f_{(i,j)} \uparrow e^p_{(x,y)}$ represents that the SFC link $e^f_{(i,j)}$ is mapped to the physical link $e^p_{(x,y)}$.

In the time after time t, CPU and bandwidth resource consumption of the SFC cannot exceed the resource capacity available in the current physical network, and the delay of physical node cannot exceed the highest delay requirement of the corresponding VNF, which is expressed as,

$$\sum_{n^f_i \in N^f} \mu^{n^p}_{n^f_i} RC_{n^f_i} \le R\_C_{n^p}, \tag{6}$$

$$\sum_{e^f_{(i,j)} \uparrow e^p_{(x,y)}} \sigma^{e^p_{(x,y)}}_{e^f_{(i,j)}} RB_{e^f_{(i,j)}} \le R\_B_{e^p_{(x,y)}}, \tag{7}$$

$$\mu_{n_i^f}^{n^p} RD_{n_i^f} \geq D_{n^p}. \tag{8}$$

In Eq. (6), $\mu_{n_i^f}^{n^p}$ is a binary variable. If the VNF n_i^f is mapped to n^p, then $\mu_{n_i^f}^{n^p} = 1$, otherwise it is equal to 0. The $\sigma_{e_{(i,j)}^f}^{e_{(x,y)}^p}$ in Eq. (7) is also a binary variable. If the SFC link $e_{(i,j)}^f$ is mapped to $e_{(x,y)}^p$, $\sigma_{e_{(i,j)}^f}^{e_{(x,y)}^p} = 1$, otherwise it is equal to 0. Eq. (8) shows that the delay value of the target physical node n^p of the VNF n_i^f cannot be greater than the highest delay requirement of n_i^f.

In addition to resource capacity constraints, SFCP also needs to follow service provision constraints. For a specific SFC, the VNFs can only be mapped to different physical nodes, i.e.,

$$\sum_{n_i^f \in N^f} \mu_{n_i^f}^{n^p} = 1, N^f \in G^f. \tag{9}$$

The service chain between adjacent VNFs may be mapped to one physical link, or it may be mapped to multiple physical links due to path division, which is defined as,

$$\sum_{e_{(i,j)}^f \in E^f} \sigma_{e_{(i,j)}^f}^{e_{(x,y)}^p} \geq 1, E^f \in G^f. \tag{10}$$

4.2 Multi Objective Optimization Model

SFCP will bring many impacts to infrastructure providers (InPs), service providers (SPs) and users. Service level agreement (SLA) play an important role in the deployment of SFC. SLA provides an agreement between SPs and user requests, or between SPs to guarantee QoS. SLA stipulates that SPs must provide users with services in accordance with service levels and performance. In the SFCP problem, SLA allows SPs to freely control network resources while guaranteeing user QoS requirements. The InP and the SP can negotiate to deploy the physical nodes required by the user request. For InPs and SPs, they want to receive as many VNFs as possible to improve revenue, so they need to improve the acceptance rate of SFC requests. For users, they want to reduce the cost of resource consumption, so they need to select the physical node with sufficient resource capacity and low unit price for mapping. All parties put forward different optimization objectives for the algorithm, and then form a multi-objective optimization problem.

For a SFC G^f, the total revenue that its successful embedding can bring to InP is calculated as follows,

$$R(G^f) = \sum_{n_i^f \uparrow n^p} P_{C_{n^p}} \times RC_{n_i^f} + \sum_{e_{(i,j)}^f \uparrow e_{(x,y)}^p} P_{B_{e_{(x,y)}^p}} \times RB_{e_{(i,j)}^f}, \tag{11}$$

where $P_{C_{n^p}}$ is the CPU resource unit price of physical node n^p and $P_{B_{e^p_{(x,y)}}}$ is the bandwidth resource unit price of physical link $e^p_{(x,y)}$. Therefore, the revenue of InP is determined by the resource demand of VNF. At the same time, the cost of resource consumption to InP is calculated as follows,

$$C(G^f) = \sum_{n_i^f \uparrow n^p} P_{C_{n^p}} \times RC_{n_i^f} + \sum_{e^f_{(i,j)} \uparrow e^p_{(x,y)}} P_{B_{e^p_{(x,y)}}} \times RB_{e^f_{(i,j)}} \times h(e^f_{(i,j)}), \tag{12}$$

where $h(e^f_{(i,j)})$ represents the number of hops of the user request link. Because in the SFC process, a user request link may require more than one physical link to provide resources.

One of the goals is to increase revenue while reducing costs, so it can be unified into revenue consumption ratio, which is defined as,

$$T_1 = \frac{R(G^f)}{C(G^f)}. \tag{13}$$

In the case of a certain unit price of resources, the means to increase the revenue of resource consumption is to increase the number of successfully embedded SFCs. Therefore, another goal is to increase the acceptance rate of SFCs, which is defined as,

$$T_2 = \lim_{T \to \infty} \frac{\sum\limits_{t=0}^{T} \lambda_{G^f}^{G^p} NUM(G^f, t)}{\sum\limits_{t=0}^{T} NUM(G^f, t)}, \tag{14}$$

among them, the binary variable $\lambda_{G^f}^{G^p}$ indicates whether the SFC G^f is successfully mapped to physical network, if it is successful, $\lambda_{G^f}^{G^p} = 1$, otherwise $\lambda_{G^f}^{G^p} = 0$. NUM represents the number of SFCs that arrive in the time range [0, t].

Therefore, our ultimate goal is to maximize the revenue consumption ratio and acceptance rate, i.e.,

$$maximize\ UT = T_1 + T_2. \tag{15}$$

5 Service Function Chain Placement Algorithm Based on Reinforcement Learning

5.1 Reinforcement Learning

Before establishing the RL model, it is necessary to clarify the various elements of RL. RL agent is the main body of algorithm training and running. We use a self-built two-layer policy network as an agent to participate in the above process. Its structure is shown in Fig. 3. In addition to the basic input layer and output

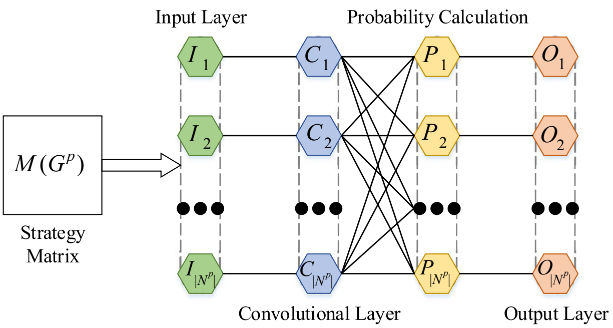


Fig. 3 Two-layer policy network model

layer, the policy network also contains two main functional layers, namely, the convolution layer and the probability calculation layer. The convolution layer evaluates the strategy vector of each physical node, and obtains the standard strategy vector format corresponding to each physical node through convolution operation. The calculation method is,

$$s\_v_{n_i^p} = \omega \times v\_n_x^p + bios, \tag{16}$$

where ω represents the weight of the convolution kernel, and *bios* represents the deviation.

The main function of probability calculation layer is to calculate the mapped probability for each physical node, in which softmax function plays an important role. The calculation method of the mapped probability of the physical node n_i^p is,

$$Prob_{n_x^p} = \frac{e^{s\_v_{n_x^p}}}{\sum_{k=1}^{|N^p|} e^{s\_v_{n_k^p}}}. \tag{17}$$

Therefore, when the SFC arrives, the two-layer policy network will perform a complete calculation and get the mapped probability and resource status of each physical node.

The environment where the agent is located is a real physical network. Only when the RL agent is trained in the physical network can it achieve good results. Therefore, we use the way of physical network information extraction to build a "real" physical network environment for agent. State refers to the current situation of physical network resource capacity. Because the arrival of SFC will occupy some network resources, the physical network state is in dynamic change. Reward signal directly affects the working mode of the agent, and then affects the performance of the algorithm. We take the total optimization target UT defined before as the reward signal of the agent. If the action taken by the agent at time t is satisfactory, it can obtain a larger UT and continue to motivate the agent to move in a better direction. Therefore, the incentive effect of reward signal and agent is positive feedback.

We build a physical network environment for RL agent by extracting necessary network resource information from the physical network. We select the four network attributes of the physical node's current available CPU resource capacity, delay value, degree and the sum of the bandwidth of the link connected to the physical node. These four attributes relate to all the elements of the algorithm optimization goal. Delay refers to the internal delay of a physical node. Only when the delay of the physical node is less than the delay requirement of the requesting node can resources be allocated. Degree is the number of links connected to the physical node. The larger the degree, the more link choices may be obtained by selecting the physical node. The sum of the bandwidth connected to the physical node refers to the sum of the available bandwidth resources of all links connected to the physical node. For the physical node n_x^p, its four corresponding network attributes are $C_{n_x^p}$, $D_{n_x^p}$, $DEG_{n_x^p}$ and $SUM\_B_{n_x^p}$ respectively. We concatenate them into a strategy vector, which is expressed as,

$$v\_n_x^p = [C_{n_x^p}, D_{n_x^p}, DEG_{n_x^p}, SUM\_B_{n_x^p}], x = 1, 2, ..., |N^p|, \tag{18}$$

where $|N^p|$ represents the total number of physical nodes. After combining the strategy vectors of all physical nodes, the strategy matrix of the physical network G^p can be obtained, which is expressed as,

$$M(G^p) = [v\_n_1^p, v\_n_2^p, ..., v\_n_{|N^p|}^p]^T = \begin{bmatrix} C_{n_1^p} & D_{n_1^p} & DEG_{n_1^p} & SUM\_B_{n_1^p} \\ C_{n_2^p} & D_{n_2^p} & DEG_{n_2^p} & SUM\_B_{n_2^p} \\ ... & ... & ... & ... \\ C_{n_{|N^p|}^p} & D_{n_{|N^p|}^p} & DEG_{n_{|N^p|}^p} & SUM\_B_{n_{|N^p|}^p} \end{bmatrix}. \tag{19}$$

When each SFC reaches the underlying network, the policy network will extract a strategy matrix from the physical network as the input, so that the RL agent can fully learn the resources of the physical network, and then make the optimal SFCP decision.

5.2 Training Process and Running Process

The purpose of training is to make RL agent adapt to the physical network environment quickly, i.e., to learn the detailed distribution of physical network resources. Only when the agent obtains stable performance in training phase, can it obtain satisfactory experimental results in the final run time. The agent always explores in the direction of the largest reward signal. We directly take the final optimization goal calculated by Eq. (15) as the reward signal of agent training. Therefore, there is always a positive feedback relationship between agent action and reward signal. We can get the mapped probability of each physical node through the calculation of the two-layer policy network. According to this probability, we arrange the physical nodes in descending order, and then map the arriving VNFs to the physical nodes in turn.

In order to reduce the loss of training accuracy, the distance between the objective function and the truly optimal strategy needs to be calculated,

$$L_{n_x^p} = -\log Prob_{n_x^p} = -\log(\frac{e^{s\_v_{n_x^p}}}{\sum_{k=1}^{|N^p|} e^{s\_v_{n_k^p}}}). \tag{20}$$

Therefore, the probability value obtained by log calculation is the softmax value of the physical node n_x^p. Then we use backpropagation to calculate the parameter gradient, which is defined as,

$$g = \alpha \times UT = \alpha \times (T_1 + T_2). \tag{21}$$

where α is the learning rate, which controls the size of the gradient and the training speed of the algorithm.

We give the complete process of RL agent training in Algorithm 1. Line 12 represents the mapping process of the VNF, and line 13 represents the mapping process of the SFC link.

Algorithm 1 Training Process

```
Input: $G^p \, G^f$;
Output: $network\ model\ parameters$;
while $iteration < epoch$ do
    for $G_i^f \in G^f$ do
        $counter = 0$;
        for $n_i^f \in N^p$ do
            $extract\ M(G^f)$;
            $s\_v_{n_x^p} = \omega \times v\_n_x^p + bios$;
            $Prob_{n_x^p} = \frac{e^{s\_v_{n_x^p}}}{\sum_{k=1}^{|N^p|} e^{s\_v_{n_k^p}}}$;
            $L_{n_x^p} = -\log Prob_{n_x^p} = -\log(\frac{e^{s\_v_{n_x^p}}}{\sum_{k=1}^{|N^p|} e^{s\_v_{n_k^p}}})$;
            $g = \alpha \times UT = \alpha \times (T_1 + T_2)$;
        end for
        if $meet\ constraints$ then
            $virtual\ network\ function\ mapping$;
            $service\ function\ chain\ link\ mapping$;
        else
            $clear\ g$;
        end if
        $counter{+}{+}$;
    end for
    $iteration{+}{+}$;
end while
$return\ parameters$;
```

According to the probability obtained by training, the VNF request in each SFC is sequentially embedded in the operation phase. After that, the shortest path algorithm is used to connect each VNF. The overall process of the operation of the MOO-SFCP algorithm based on RL is shown in Algorithm 2. Line 2 is the mapping process of the VNFs, and the line 4 is the mapping process of the SFC links.

Algorithm 2 Running Process

```
Input: $G^p$ $G^f$;
Output: $resource\ consumption\ revenue$, $resource\ consumption\ cost$, $VNF\ acceptance\ rate$;
for $G_i^f \in G^f$ do
    $VNF\ mapping\ based\ on\ probability$;
end for
$shortest\ path\ algorithm$;
if $isMapped(\forall G^f)$ then
    $return\,(success)$;
end if
```

6 Performance Evaluation

6.1 Experimental Setup

We use Pycharm 2018 to build a Tensorflow-based platform to simulate simulation experiments [37, 38]. All experiments are performed on a Windows 8 system equipped with Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz.

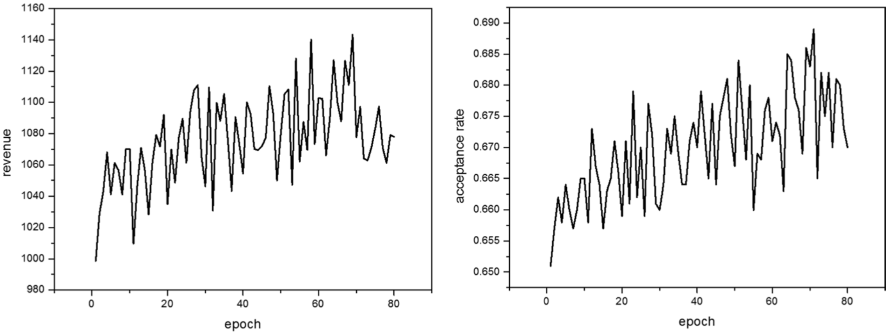
We generate a text file through CodeBlocks programming to save the relevant information of physical network [39], including key information such as the number of physical nodes, node connection relationships, node resource capacity and attributes, link resource capacity and other key information. In addition, we have also generated 2,000 SFC request files through programming simulation, and the related information of the SFC is also stored in these files. Among them, 1,000 are used as the training set and the rest are used as the test set. The main information contained in the SFC file includes the number of SFC requests, the connection relationship between virtual network requests, the number of resource requests for VNFs, and the number of resource requests for SFCs. The detailed experimental parameter settings are shown in Table 3.

Table 3 Parameter setting

Parameter	Value	Minimum	Maximum
Physical nodes	100	–	–
Physical links	600	–	–
Physical CPU resource	–	50 Gflops	100 Gflops
Physical delay attribute	–	1 ms	10 ms
Physical bandwidth attribute	–	50 Mbps	100 Mbps
SFC requests	2000	–	–
VNF requests	–	2	10
VNF CPU request	–	1 Gflops	20 Gflops
VNF delay request	–	1 ms	10 ms
VNF link bandwidth request	–	1 Mbps	20 Mbps

Table 4 Algorithm Comparison

Algorithm	Description
MOO-SFCP	Consider the optimization of multiple goals such as resource revenue, cost, service function request, etc. The SFC is modeled as MDP. The mapping probability of physical nodes is deduced by using the RL model of double-layer policy network, and the link mapping of SFC is completed according to the shortest path algorithm
RLVNE [40]	The neural network model is trained by using the historical data of virtual network request, and the mapping probability of each physical node is derived by using the strategy gradient training method of automatic optimization. The virtual link mapping is completed by using the breadth first strategy
Baseline [41]	The formula $H(n^p) = CPU(n^p) \sum_{e^p \in E^p} B(E^p)$ is used to sort the underlying network nodes, the availability concept of the underlying nodes is defined, and the shortest path algorithm is used to complete the virtual link mapping process



(a) Training result of resource consumption revenue

(b) Training result of VNF request acceptance rate

Fig. 4 Training results

We use the RLVNE algorithm proposed in [40] and the Baseline algorithm proposed in [41] as the main comparison algorithms. The former is a virtual network embedding (VNE) algorithm based on RL, and the latter is a heuristic VNE algorithm based on node ranking. Because the essence of VNE algorithm is also the allocation of network resources, they can be used as contrast algorithms for MOO-SFCP algorithm based on RL [42]. The above two comparison algorithms participate in the comparison from two different perspectives of ML and heuristic, so they are more comprehensive. We summarize the related information of the MOO-SFCP algorithm based on RL and the above two algorithms in Table 4.

6.2 Experimental Results and Analysis

In order to illustrate the effectiveness of the two-tier policy network training, we test the resource revenue generated by the SFCP and the request acceptance rate of the SFC. The experimental results are shown in Fig. 4. We conduct 80 epoch training on the training set containing 1,000 SFC request files. Figure 4a and b show the training changes in resource revenue and acceptance rate, respectively. In the first 20 epochs, since the agent has added a new training environment, it is unfamiliar with the surrounding resource attribute status, so its performance is not stable at this time. In the middle of training, the agent adapts to the environment after a period of learning, and some actions taken may get positive reward signals, so the performance of the agent tends to stabilize at this time. In the last 20 epochs, the agent's performance is stable within a certain range. On the one hand, the agent thoroughly understands the underlying network environment, and long term learning enables it to take actions that are beneficial to itself. On the other hand, because the complexity of the problem that the two-layer policy network can handle is limited, when the training is long enough, the performance of the agent reaches its limit.

In the algorithm operation stage, directly use the training results to perform the SFCP operation. Figures 5 and 6 respectively show the performance comparison between the RL-based MOO-SFCP algorithm and the other two algorithms in terms of resource revenue and request acceptance rate. First of all, the overall trend of the three algorithms is gradually decreasing. In the early stage of algorithm operation, the physical network resources are relatively sufficient, and most of the SFC requests reached at this time can obtain the required service resources, so the request acceptance rate at this time is relatively high. Due to the large number of successfully embedded VNFs at this time, InPs and SPs will obtain more substantial revenue. The performance of the following three algorithms all show a downward trend, and the reduction in the number of available physical resources is the direct cause. The performance of the three algorithms has obvious differences. This is because our algorithm not only combines with RL, but also pays more attention to the multi-objective optimization process, so it is better than the other two algorithms in terms of specific performance.

Figure 7 shows the comparison of the three algorithms in terms of resource revenue-cost ratio. Our algorithm has achieved better results than the other two

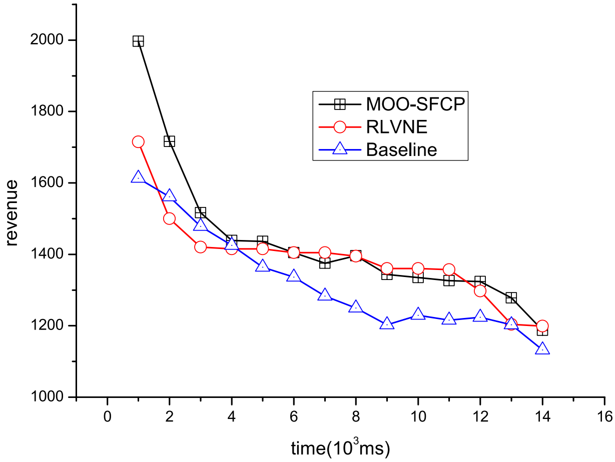


Fig. 5 Resource consumption revenue

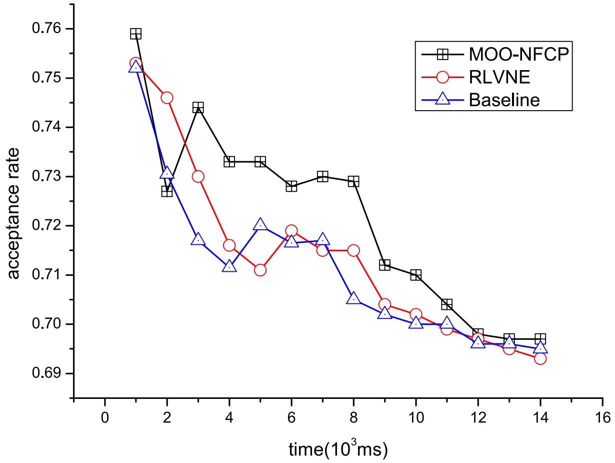


Fig. 6 VNF request acceptance rate

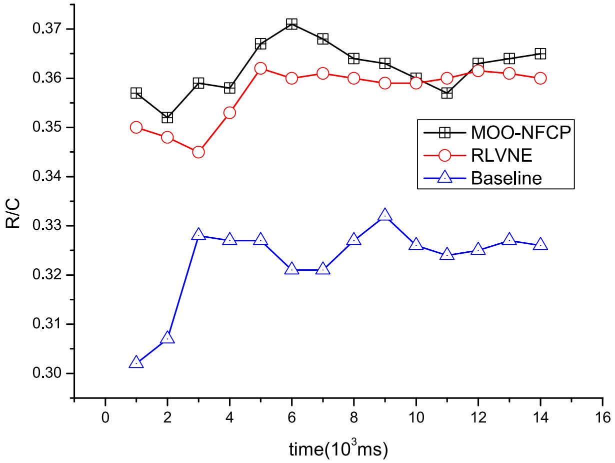


Fig. 7 Revenue cost ratio

algorithms. The revenue-cost ratio reflects the degree of utilization of physical network resources. We take the two-layer policy network as an intelligent agent to participate in the training. The training environment is composed of physical resource conditions. Therefore, the agent can obtain better training results and then make the optimal resource allocation strategy. The experimental results also show that our algorithm has advantages over general resource allocation algorithms based on ML and heuristic methods.

We try to explore the possible impact on algorithm performance by changing the demand attributes of VNFs. Specifically, the CPU resource demand and bandwidth demand requested by the SFC remain unchanged. We set the delay attribute value of

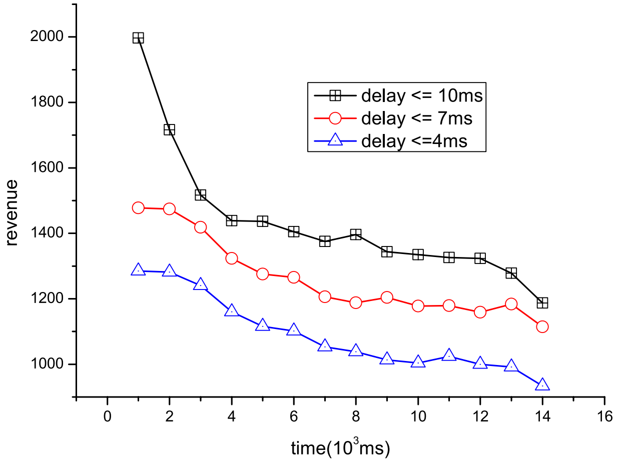


Fig. 8 Resource consumption revenue in different delay demand

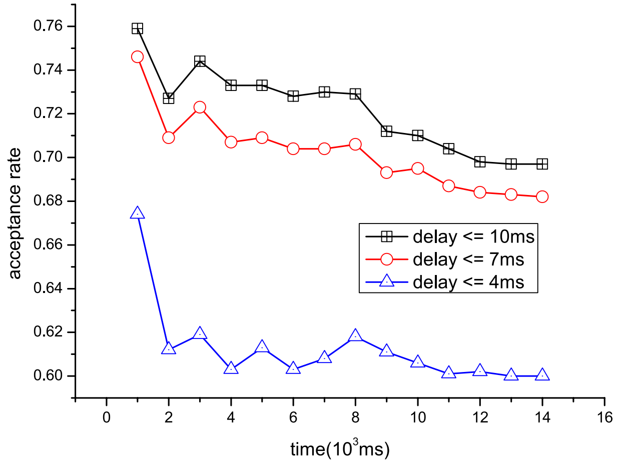


Fig. 9 VNF request acceptance rate in different delay demand

the VNF to be less than or equal to 10 ms, less than or equal to 7 ms, and less than or equal to 4 ms. The changes in resource consumption revenue and request acceptance rate under extended demand conditions are shown in Figs. 8 and 9, respectively.

It can be seen that with the decrease of VNF delay demand level, the resource consumption revenue and request acceptance rate of the algorithm decrease as a whole. Because when the delay requirement of VNF request is reduced, the number of physical nodes that can meet the mapping conditions will be reduced, and the number of SFCs that can be successfully embedded in the physical network will be reduced, so the experimental results are in line with the expected and actual situation.

The difference of resource revenue cost ratio under different delay requirements is obvious (Fig. 10). Because the revenue cost ratio is determined by the revenue and cost of network resource allocation, it will not show an obvious upward or downward trend due to the change of available resources. When the delay demand is high, the number of qualified physical nodes decreases, and the revenue and cost of resource allocation are reduced, but the revenue cost ratio is relatively low. When the delay demand is low, the number of qualified physical nodes increases, and the revenue and cost of resource allocation increase, but the revenue cost ratio is relatively high. Therefore, it can be shown that the MOO-SFCP algorithm based on RL has certain flexibility in dealing with the changes of network environment.

7 Conclusion

As a new type of future network architecture, virtual network architecture has broad application prospects. To a large extent, the function requests of network users depend on the effective allocation of virtual network resources. The essence of SFCP is the allocation process of physical network resources, and it can effectively cope with dynamic changes of physical resources and complex network topologies. Combined with the latest developments in ML, we model the SFCP as a MDP, and propose a MOO-SFCP algorithm based on RL. Based on the realization of the basic VNF request mapping, the optimization of multiple algorithm goals is realized. In RL algorithm, we use a self-defined two-layer policy network as an agent, and the information matrix extracted from the physical network is used as the agent's training environment. The purpose is to obtain the mapped probability of each physical node. The SFCP is completed according to the probability. The final simulation experiment verify the excellent performance of the algorithm from multiple

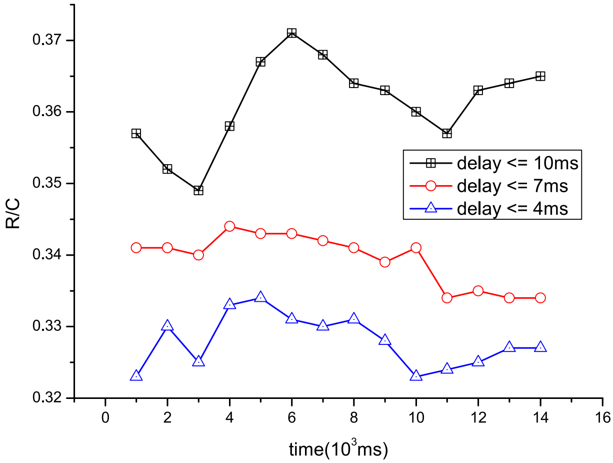


Fig. 10 VNF revenue cost ratio in different delay demand

indicators of revenue, cost and acceptance rate. In addition, we also explore the impact of the change of the delay attribute on the flexibility of the algorithm.

As part of our future work, we will continue to explore the possible role of different ML models in the field of SFCP. In addition, we also plan to deeply mine the resource information of the physical network, deduce the internal relationship between different resource attributes, and explore the impact of extracting other resource attributes on the algorithm. Last but not least, it is necessary to conduct in-depth research on using vertical solutions to solve SFCP problems.

References

1. Zhang, P., Wang, C., Jiang, C., Han, Z.: Deep reinforcement learning assisted federated learning algorithm for data management of IIoT. IEEE Trans. Ind. Inform. **17**(12), 8475–8484 (2021)
2. Wang, X., Ma, R.T.B.: On the Tussle between over-the-top and internet service providers: analysis of the Netflix-Comcast type of deals. IEEE/ACM Trans. Netw. **28**(6), 2823–2835 (2020)
3. Key, P., Steinberg, R.: Pricing, competition and content for internet service providers. IEEE/ACM Trans. Netw. **28**(5), 2285–2298 (2020)
4. Zhang, P., Wang, C., Aujla, G.S., Kumar, N., Guizani, M.: IoV scenario: implementation of a bandwidth aware algorithm in wireless network communication mode. IEEE Trans. Veh. Technol. **69**(12), 15774–15785 (2020)
5. Cherrared, S., Imadali, S., Fabre, E., Gössler, G., Yahia, I.G.B.: A survey of fault management in network virtualization environments: challenges and solutions. IEEE Trans. Netw. Serv. Manag. **16**(4), 1537–1551 (2019)
6. Liu, Y., Lu, H., Li, X., Zhao, D.: An approach for service function chain reconfiguration in network function virtualization architectures. IEEE Access **7**, 147224–147237 (2019)
7. Yu, Y., Bu, X., Yang, K., Nguyen, H.K., Han, Z.: Network function virtualization resource allocation based on joint benders decomposition and ADMM. IEEE Trans. Veh. Technol. **69**(2), 1706–1718 (2020)
8. Wang, C., Batth, R.S., Zhang, P., Aujla, G.S., Duan, Y., Ren, L.: VNE solution for network differentiated QoS and security requirements: from the perspective of deep reinforcement learning. Computing **103**(6), 1061–1083 (2021)
9. Thiruvasagam, P.K., Chakraborty, A., Mathew, A., Murthy, C.S.R.: Reliable placement of service function chains and virtual monitoring functions with minimal cost in softwarized 5G networks. IEEE Trans. Netw. Serv. Manag. **18**(2), 1491–1507 (2021)
10. Yin, X., Cheng, B., Wang, M., Chen, J.: Availability-Aware Service Function Chain Placement in Mobile Edge Computing (vol. 2020, pp. 69–74). IEEE World Congress on Services (SERVICES). Beijing, China (2020). https://doi.org/10.1109/SERVICES48979.2020.00028
11. Dieye, M., Ahvar, S., Sahoo, J., Ahvar, E., Glitho, R., Elbiaze, H., Crespi, N.: CPVNF: cost-efficient proactive VNF placement and chaining for value-added services in content delivery networks. IEEE Trans. Netw. Serv. Manag. **15**(2), 774–786 (2018)
12. Zhang, P., Wang, C., Jiang, C., Benslimane, A.: Security-aware virtual network embedding algorithm based on reinforcement learning. IEEE Trans. Netw. Sci. Eng. **8**(2), 1095–1105 (2020)
13. Wei, X., Zhao, J., Zhou, L., Qian, Y.: Broad reinforcement learning for supporting fast autonomous IoT. IEEE Internet Things J. **7**(8), 7010–7020 (2020)
14. Lee, H., Cha, S.W.: Reinforcement learning based on equivalent consumption minimization strategy for optimal control of hybrid electric vehicles. IEEE Access **9**, 860–871 (2021)
15. Li, D., Hong, P., Xue, K., Pei, J.: Virtual network function placement and resource optimization in NFV and edge computing enabled networks. Comput. Netw. **152**, 12–24 (2019)
16. Li, D., Hong, P., Xue, K., Pei, j: Virtual network function placement considering resource optimization and SFC requests in cloud datacenter. IEEE Trans. Parallel Distrib. Syst. **29**(7), 1664–1677 (2018)
17. Qi, D., Shen, S., Wang, G.: Towards an efficient VNF placement in network function virtualization. Comput. Commun. **138**, 81–89 (2019)

18. Tang, H., Zhou, D., Chen, D.: Dynamic network function instance scaling based on traffic forecasting and VNF placement in operator data centers. IEEE Trans. Parallel Distrib. Syst. **30**(3), 530–543 (2019)
19. Hawilo, H., Jammal, M., Shami, A.: Network function virtualization-aware orchestrator for service function chaining placement in the cloud. IEEE J. Sel. Areas Commun. **37**(3), 643–655 (2019)
20. Li, J., Shi, W., Wu, H., Zhang, S., Shen, X.: Cost-aware dynamic SFC mapping and scheduling in SDN/NFV-enabled space-air-ground integrated networks for internet of vehicles. IEEE Internet Things J **1**, 1–15 (2020). https://doi.org/10.1109/JIOT.2021.3058250
21. Ghazizadeh, A., Akbari, B., Tajiki, M.M.: Joint reliability-aware and cost efficient path Allocation-Fig and VNF placement using sharing scheme. J. Netw. Syst. Manag. **30**(1), 5 (2022)
22. Pei, J., Hong, P., Xue, K., Li, D.: Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system. IEEE Trans. Parallel Distrib. Syst. **30**(10), 2179–2192 (2019)
23. Pei, J., Hong, P., Xue, K., Li, D.: Resource aware routing for service function chains in SDN and NFV-enabled network. IEEE Trans. Serv. Comput. **14**(4), 985–997 (2021)
24. Pei, J., Hong, P., Xue, K., Li, D., Wei, D.S.L., Wu, F.: Two-phase virtual network function selection and chaining algorithm based on deep learning in SDN/NFV-enabled networks. IEEE J. Sel. Areas Commun. **38**(6), 1102–1117 (2020)
25. Bari, F., Chowdhury, S.R., Ahmed, R., Boutaba, R., Duarte, O.C.M.B.: Orchestrating virtualized network functions. IEEE Trans. Netw. Serv. Manag. **13**(4), 725–739 (2016)
26. Mechtri, M., Ghribi, C., Zeghlache, D.: A scalable algorithm for the placement of service function chains. IEEE Trans. Netw. Serv. Manag. **13**(3), 533–546 (2016)
27. Pham, C., Tran, N.H., Ren, S., Saad, W., Hong, C.S.: Traffic-aware and energy-efficient vNF placement for service chaining: joint sampling and matching approach. IEEE Trans. Serv. Comput. **13**(1), 172–185 (2020)
28. Santos, G.L., Bezerra, D.D.F., Rocha, D.É.S., Ferreira, L., Moreira, A.L.C., Gonçalves, G.E., Marquezini, M.V., Recse, Á., Mehta, A., Kelner, J., Sadok, D., Endo, P.T.: Service function chain placement in distributed scenarios: a systematic review. J. Netw. Syst. Manag. **30**(1), 1–39 (2022)
29. Mahboob, T., Jung, Y.R., Chung, M.Y.: Dynamic VNF placement to manage user traffic flow in software-defined wireless networks. J. Netw. Syst. Manag. **28**(3), 436–456 (2020)
30. Pei, J., Hong, P., Pan, M., Liu, J., Zhou, J.: Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks. IEEE J. Sel. Areas Commun. **38**(2), 263–278 (2020)
31. Quang, P.T.A., Hadjadj-Aoul, Y., Outtagarts, A.: A deep reinforcement learning approach for VNF forwarding graph embedding. IEEE Trans. Netw. Serv. Manag. **16**(4), 1318–1331 (2019)
32. Sun, P., Lan, J., Li, J., Guo, Z., Hu, Y.: Combining deep reinforcement learning with graph neural networks for optimal VNF placement. IEEE Commun. Lett. **25**(1), 176–180 (2021)
33. Li, B., Lu, W., Liu, S., Zhu, Z.: Deep-learning-assisted network orchestration for on-demand and cost-effective VNF service chaining in inter-DC elastic optical networks. IEEE/OSA J. Opt. Commun. Netw. **10**(10), D29–D41 (2018)
34. Troia, S., Alvizu, R., Maier, G.: Reinforcement learning for service function chain reconfiguration in NFV-SDN metro-core optical networks. IEEE Access **7**, 167944–167957 (2019)
35. Wang, T., Zu, J., Hu, G., Peng, D.: Adaptive service function chain scheduling in mobile edge computing via deep reinforcement learning. IEEE Access **8**, 164922–164935 (2020)
36. Xiao, Y., Zhang, Q., Liu, F., Wang, J., Zhao, M., Zhang, Z., Zhang, J.: NFVdeep: adaptive online service function chain deployment with deep reinforcement learning. In: 2019 IEEE/ACM 27th international symposium on quality of service (IWQoS)), pp. 1–10 (2019)
37. Hu, Q., Ma, L., Zhao, J.: DeepGraph: a PyCharm tool for visualizing and understanding deep learning models. In: 2018 25th Asia-Pacific software engineering conference (APSEC)), pp. 628–632 (2018)
38. Suen, H., Hung, K., Lin, C.: TensorFlow-based automatic personality recognition used in asynchronous video interviews. IEEE Access **7**, 61018–61023 (2019)
39. Santiago Iii, J. M., Nodalo, G., Valenzuela, J., Deja, J. A.: Explore, edit, guess: understanding novice programmers' use of codeblocks for regression experiments. In: CEUR workshop proceedings, vol. 3054, pp. 3–17 (2021)
40. Yao, H., Chen, X., Li, M., Zhang, P., Wang, L.: A novel reinforcement learning algorithm for virtual network embedding. Neurocomputing **284**, 1–9 (2018)
41. Yu, M., Yi, Y., Rexford, J., Chiang, M.: Rethinking virtual network embedding: substrate support for path splitting and migration. ACM Sigcomm Comput. Commun. Rev. **38**(2), 17–29 (2008)

42. Zhang, P., Yao, H., Liu, Y.: Virtual network embedding based on computing, network, and storage resource constraints. IEEE Internet Things J. **5**(5), 3298–3304 (2018)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Hongtai Liu Hongtai Liu graduated from China University of Petroleum (East China) and received his master's degree in 2014. He is working in CNPC Research Institute of Safety and Environmental Technology. His research interests include big data analysis, artificial intelligence, software quality and reliability.

Shengduo Ding Shengduo Ding graduated from China University of Petroleum (East China) and received his master's degree in 2020. He is working in CNPC Research Institute of Safety and Environmental Technology. His research interests include computer version and machine learning.

Shunyi Wang Shunyi Wang graduated from Jilin University and received his master's degree in 2007. He is working in CNPC Research Institute of safety and Environmental Technology. His research interests include information management, environmental engineering and software engineering.

Gang Zhao Gang Zhao graduated from Xi'an University of Petroleum and received his master's degree in 2006. He is working in CNPC Research Institute of safety and Environmental Technology. His research interests include big data analysis, artificial intelligence, software quality and reliability.

Chao Wang Chao Wang is a graduate student in the College of Computer Science and Technology, China University of Petroleum (East China). His research interests include network artificial intelligence, network virtualization and wireless network.

Authors and Affiliations

Hongtai Liu[1] · Shengduo Ding[1] · Shunyi Wang[1] · Gang Zhao[1] · Chao Wang[2]

Hongtai Liu
liuhongtai@cnpc.com.cn

Shengduo Ding
dingshengduo@cnpc.com.cn

Shunyi Wang
wangshunyi@cnpc.com.cn

Gang Zhao
derek@cnpc.com.cn

[1] CNPC Research Institute of Safety and Environment Technology, Beijing 102206, China

[2] College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266580, China