Cluster Computing (2024) 27:7657–7671
https://doi.org/10.1007/s10586-024-04335-4

Energy-delay-aware VNF scheduling: a reinforcement learning approach with hierarchical reward enhancement

Xi-Wei Yang[1] · Xiao-Chun Wu[1] · Yue Shao[1] · Gan Tang[1]

Received: 11 December 2023 / Revised: 17 January 2024 / Accepted: 1 February 2024 / Published online: 29 March 2024

Abstract
Network functions virtualization (NFV) is a technology that virtualizes network functions into virtual network functions (VNF) to deliver communication services. Efficient and flexible VNF scheduling is an important way to improve network resource utilization, reduce costs, and provide better service quality. With the development of artificial intelligence, network equipment is no longer just a forwarding node, but also a computing node. Therefore, energy consumption will become an important indicator that needs to be considered in the VNF scheduling problem. In this paper, we aim to realize VNF scheduling with minimizes idle energy loss (IEL) of NFV nodes and the makespan (i.e., overall completion time) for all services. The problem can be formulated as a Mixed Integer Linear Program (MILP), and the complexity of the problem grows exponentially as the size of the network scale increases. To solve this problem efficiently and flexibly, we treat MILP as a Markov Decision Process (MDP) and design an reinforcement learning (RL) algorithm to solve the MDP problem. Specifically, the algorithm utilizes a hierarchical reward enhancement (HRE) mechanism, called RL-HRE. In addition, a weighted reward function is carefully designed in the proposed algorithm to achieve flexible energy-delay-aware VNF scheduling. The simulation results show that RL-HRE is superior to other comparative algorithms in terms of solution accuracy and time complexity.

Keywords Network functions virtualization · VNF scheduling · Mixed integer linear program · Reinforcement learning

1 Introduction

As key enabling technologies of future networks, software defined networking (SDN) and network function virtualization (NFV), two complementary technologies, have inherent advantages in flexibility and cost reduction compared with traditional network service technologies [1, 2]. SDN decouples the network into a control plane and a data plane, allowing network control to achieve programmability and underlying infrastructure to be abstracted from network services. NFV abstracts network functions (such as network address translation, firewall, etc.) that originally needed to be implemented using dedicated hardware devices, these network functions can be implemented through virtualization technology, such as virtual machines, containers, and Commercial Off-The-Shelf (COTS) servers, etc. These software-based network functions called virtual network functions (VNF) are orchestrated and linked together to form service function chains (SFC) to support network services requirements. Different SFCs composed of different numbers and different sequences of VNFs are used to support different network service requirements. This software-based implementation of network functions brings many benefits, which we foresee include: fast network equipment migration management, greatly reducing equipment costs, and achieving high utilization and low energy consumption of physical resources through aggregation. With the explosive growth of network service demand and network service flexibility requirements, integrating these two technologies into future networks can significantly reduce network resource costs and improve network flexibility.

✉ Xi-Wei Yang
21020090044@pop.zjgsu.edu.cn

Xiao-Chun Wu
spring-403@zjgsu.edu.cn

[1] Department of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou 310020, Zhejiang, China

How to rationally allocate physical resources in the underlying network to realize the embedding of multiple SFC to achieve efficient and agile network services is called NFV resource allocation (NFV-RA), it includes VNF composition, VNF embedding (i.e., the process of embedding multiple VNFs and links on NFV infrastructure) and VNF scheduling embedded on NFV nodes [3]. NFV-RA necessitates efficient algorithms to determine VNF chain composition, mapping VNFs and links into appropriate location, and optimize VNF schedule tasks. Otherwise, it will cause many problems such as low utilization of physical resources and inability to efficiently meet customer service needs. These three parts are sequential processes. First, it is necessary to determine VNFs chain composition, then embed VNFs chain into the underlying network infrastructure, including NFV nodes and links, and finally determine the scheduling sequence of packet processing by the underlying VNFs located on NFV nodes. The scheduling framework depicted in Fig. 1 of Sect. 3 provides a concise overview of this process. For the first two parts, researchers have explored many mathematical models aiming at different optimization objectives or considering various related factors. Under the premise of VNF embedding, how to efficiently and flexibly schedule all embedded VNFs to ensure network service support is an important challenge of NFV-RA. The existing studies have shown that the classic VNF scheduling problem can be regarded as a job shop scheduling problem (JSP) [4, 5], which is an NP-hard combinatorial optimization problem [6]. The job shop problem is a well-known issue in the fields of computer science and operation research and has to find time slots for a series of activities according to different objectives under given constraints, such as precedence constraints between activities, resource constraints, etc [7]. Makespan is the most classic objective of this problem, which refers to the duration from project initiation to completion. Hence, the VNF scheduling problem has a resemblance to it about delay issue, its objective is to schedule all VNFs embedded on NFV nodes to minimize makespan (i.e., the time period from the execution of the first VNFs flow packet to the completion of the last VNFs flow packet among all the scheduled VNFs for all the network services). The advancement of artificial intelligence technology has brought about significant transformations in the virtualization technology of network equipment, which serves as a provider of network services for artificial intelligence computing power. Network equipment is no longer solely responsible for information transmission; it now actively participates in artificial intelligence computing tasks [8]. Consequently, energy consumption has emerged as a crucial metric to gauge network performance during the establishment of bearer networks following network function virtualization. However, considering that little research has been conducted on VNF scheduling and existing research on VNF scheduling has primarily focused on delay issues while neglecting the energy consumption due to the operational characteristics of NFV nodes in real-world scenarios. Therefore, it is imperative to incorporate energy loss factors that reflect the actual operating conditions of NFV nodes into the VNF scheduling process. To obtain its approximate solution, a large number of heuristics

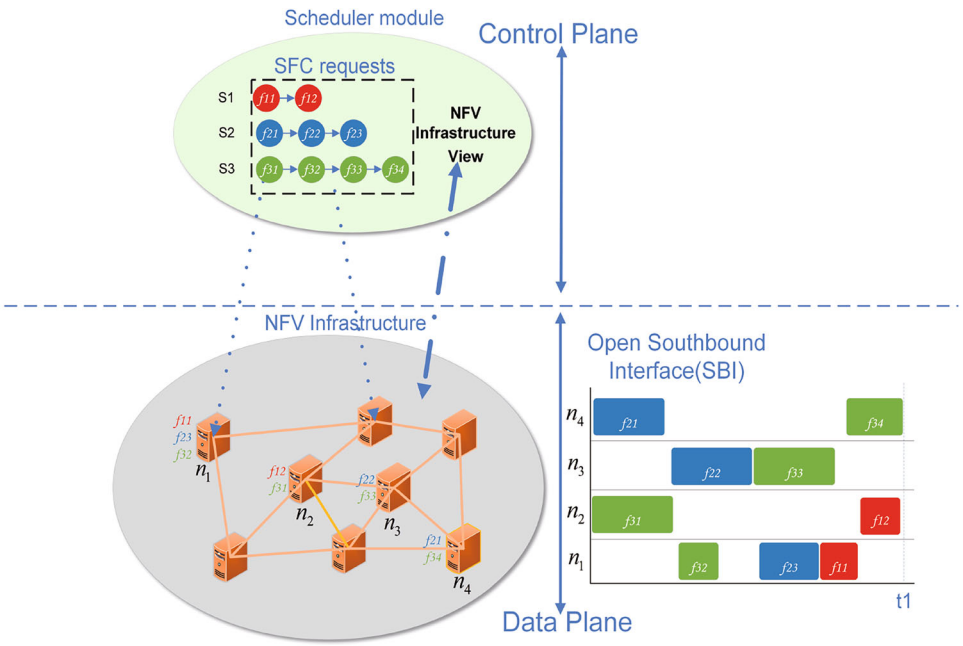


Fig. 1 An SDN/NFV-enabled VNFs scheduling framework

algorithm (e.g., greedy algorithm) and meta-heuristic algorithm (e.g., genetic algorithm (GA)) have been developed [9–12], but these methods usually have many disadvantages, such as the highly dependent characteristic of the solution performance and the low convergence speed.

Reinforcement learning is a promising approach to solving combinatorial optimization problems in terms of guaranteed accuracy with lower computational complexity [13]. At the same time, the VNF scheduling process aims to determine the precise timing for initiating data packet processing by a series of VNFs deployed on each NFV node, while ensuring compliance with relevant constraints. It can be perceived as a sequential decision-making procedure, where decisions are made at each moment (i.e., determine the required VNF responsible for packet processing at the beginning of each time slot) until all VNF decision-making tasks are completed. By progressively determining decisions at each step, an overall scheduling plan is ultimately established. This sequential decision-making process aligns with dynamic programming principles (breaking down decisions into a sequence of steps over time to simplify them), and MDP modeling is frequently employed in solving dynamic programming problems, hence, the VNF scheduling process can be regarded as a MDP. Therefore, we use the RL algorithm to learn the best scheduling strategy through continuous exploration, which leads to significant time benefits in schedule decisions while ensuring good precision.

The contribution of this paper is three-fold. First, in view of the fact that previous research generally did not consider the energy consumption issue in the VNF scheduling process and the network nodes bear the computing background, we introduce the MILP model based on existing research, which aims to minimize the weighted sum of idle energy consumption and scheduling delay. Second, we improved previous research with a new state representation method and reward update mechanism (i.e. hierarchical reward enhancement) for Q-learning and the proposed algorithm is called RL-HRE. Third, a weighted reward function is designed in the proposed algorithm to implement delay-energy flexible scheduling. In addition, we compared the convergence performance of the RL-HRE and the original algorithm [14] (called RL-Primitive), and conducted multiple experiments to verify that the proposed reward function effectively implemented flexible energy-delay-aware scheduling. Finally, the performance of relevant indicators is compared between RL-HRE and some comparative algorithms under scheduling instances of different sizes. The simulation results show the superiority of the proposed algorithm in general.

The rest of the article is organized as follows. Section 2 discusses related studies. Section 3 introduces in detail the VNF scheduling process under the SDN/NFV network architecture, the VNF scheduling energy loss model we introduce, and the proposed MILP model. Section 4 describes the details of the proposed RL-HRE scheme. Section 5 reports the simulations and results. Lastly, Sect. 6 concludes the article.

2 Related work

Compared to the extensive research on the VNF embedding problem, the VNF scheduling problem is still in its infancy. For the first time, Riera et al. [4] proposed a formal model of the VNF complex scheduling problem, dividing the problem into two phases: the server allocates the corresponding VNFs and then resolves the resulting JSP. Mijumbi et al. [15] proposed four simple algorithms for the VNF scheduling problem as the basis for future research in this field, these algorithms are evaluated on six indicators but energy consumption considerations are ignored. Qu et al. [10] formulated the VNF scheduling traffic steering problem as MILP, the goal is to minimize the running time of all VNFs scheduling under the premise of meeting strict service delay requirements and developed a genetic algorithm to solve this problem. Pham et al. [12] formulate joint operational (consumption cost and active power cost) and network traffic (joint traffic rate requirements and hop distance) problem based on VNFs placement constraints and cost models, using sampling-based markov approximation and the matching method to solve it. The gap between the solution obtained by this framework and the optimal solution is close. Promwongsa et al. [16] formulated the joint VNF placement and scheduling problem for delay-sensitive network services as ILP and proposed two efficient heuristic algorithms and an algorithm based on tabu search to solve it. Li et al. [14] consider the delay requirements of each service in the VNF scheduling process, proposed its MILP model and developed a reinforcement learning algorithm with a variable action set, which outperformed other benchmark algorithms in terms of performance and computational efficiency. An algorithm [17] is proposed to maximize the number of accepted chains by balancing edge and cloud resource utilization. It focuses VNF placement strategy aimed at minimizing energy loss on the physical substrate. Most VNF scheduling research focuses on scheduling delay [10, 14–17], and energy-saving scheduling research still needs to be explored. In addition, these studies [12, 17, 18] all consider the issue of energy loss, but they focus on the energy consumption issue about the VNF embedding process. These studies usually establish models from the spatial perspective of energy consumption, such as evaluating energy usage based on the resource utilization of physical equipment and using the characteristics of

physical equipment (embedded device computing units or high-performance servers) as criteria for assessing energy consumption. However, modeling from a temporal perspective in the energy consumption model can better evaluate the economic efficiency of the solution over a specific period of time. The VNF scheduling process follows the time slot operation characteristics of the device (i.e., its power consumption changes according to different time slots), and in the context of network nodes participating in artificial intelligence computing, so it is necessary to conduct energy consumption modeling from scheduling perspective.

In view of the existing research on energy-aware scheduling of JSP [19, 20] and the operating characteristics of NFV nodes [21], we introduce idle energy loss (IEL) to simulate the NFV node scheduling energy consumption in the scheduling process, this study focuses on energy-saving factors and tries to enrich the research in this field.

3 System architecture and problem modeling

To illustrate the VNF scheduling process, a VNF scheduling network architecture under the background of SDN/NFV is proposed by Li et al. [14]. As shown in Fig. 1, the control plane is separated from the data plane, where the NFV infrastructure on the data plane arranges the VNFs belonging to different SFC embedded on the NFV nodes through the scheduling decision information from the control plane, and configures the relevant routing information to realize the traffic processing of different network service packages need. The VNF scheduler located on the control plane simultaneously schedules multiple Service Requests (SR) in the same batch at the beginning of each scheduling cycle, and each batch SR needs to be supported by multiple SFC composed of different orders and different numbers of VNFs to meet various network service requirements. Notice that the consideration of bandwidth and resource requirements is unnecessary for scheduling problem, with the sole focus being on the processing sequence of VNFs. After the VNF control plane receives the SFC request and forms the VNF chains, the corresponding VNF is embedded into the NFV nodes(i.e. the orange device in Fig. 1) located in the data plane. Then the VNF scheduler module located in control plane solves the VNF scheduling scheme, that is, determines the time for VNFs located on the NFV nodes to perform packet processing. As shown in Fig. 1, three SFC requests form three VNF chain, all the required VNF is embedded in the four NFV nodes (note that the embedding process is not the focus of this paper), and then the scheduling scheme at the bottom right is determined where $t1$ in the figure is the makespan of this scheduling scheme. At the beginning of a time slot of each scheduling cycle, the status information of the NFV nodes is collected by the VNF scheduler module through the open southbound interface, and the collected status information is used by the VNF scheduler to make the desired VNF schedule decisions to the data plane. This operation is repeated until the end of the entire scheduling cycle to complete the construction of all SFC to meet network service traffic processing requirements. If the batch of SR in the next scheduling cycle remain unchanged, then the scheduling decision made by the VNF scheduler in the previous scheduling cycle can be reused. If the SR (e.g., the length of the required SFC, the precedence relations of the VNFs, etc.) changes, then the VNF scheduler needs to recalculate to obtain the optimal scheduling scheme.

3.1 Energy-efficient scheduling

In the process of network technology trends in the future, people will pay more and more attention to energy-saving networks. Energy-saving JSP research is divided into three categories based on saving methods [22]: 1. Reduce energy consumption during machine operation; 2. Reduce idle time during machine operation; 3. Reduce unnecessary settings or startups during machine operation. In this paper, we use the second energy-saving approach to formulate the energy consumption problem of VNF scheduling. For the convenience of modeling, we assume that all NFV nodes meet certain energy-saving specifications (i.e., the NFV nodes automatically shut down after processing the data packets that need to be processed within a scheduling cycle), then the idle time of NFV nodes in a scheduling cycle is the time when the VNFs on it are not executing (i.e. no packet processing is performed or perform computing tasks). Figure 2 shows two scheduling schemes $(a)(b)$ with the same makespan (if t1 = t2), according to the above energy-saving specification, if the energy consumption each time slot of VNFs on two NFV nodes is the

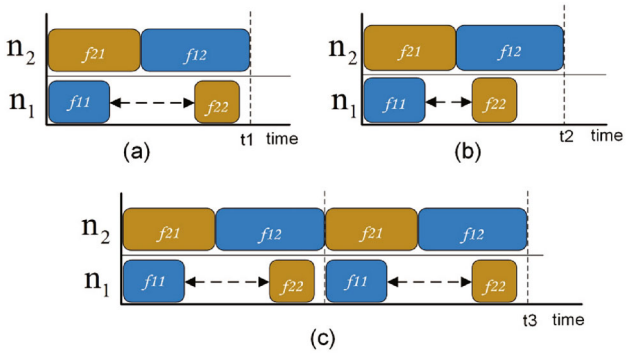


Fig. 2 Energy-efficient scheduling model explanation

same, the arrows indicate IEL on the NFV nodes, and it is obvious that the IEL of (*a*) scheme is larger. When a large number of packets with the same SFC requirements need to be processed, this scheduling execution scheme will periodically affect the device power consumption in accordance with the clock frequency within the device. As shown in Fig. 2c, this is a schematic diagram of using scheduling plan (*a*) and requiring two scheduling cycles for packet processing, Where total IEL for processing these packets is two times IEL of (*a*), IEL increases multiple times with the scheduling period required for packet processing, and the energy economy of the lower IEL scheduling scheme will be more prominent. Therefore, for a lower IEL, we need to make the energy consumption of a scheduling cycle scheme as low as possible.

3.2 MILP model

In this section, we add the above energy-saving scheduling model to energy-delay-aware MILP modeling, the IEL of the scheduling scheme is designed as an equality constraint and the other constraints refer to existing literature [23]. In addition, the scheduling solution goal is changed from minimizing the makespan only to minimizing the weighted sum of IEL and makespan. Each NFV node supports embedding multiple VNFs, but the computing resources of each NFV node can only be utilized by one VNF for packet batch processing at a time [1, 3, 10], where the makespan of the VNF scheduling problem is defined as the duration of scheduling all embedded VNFs to process all single packets of the same batch of SR once. Similarly, according to the above energy-saving model, the IEL of a VNF scheduling cycle is the sum of the idle time energy consumption of all NFV nodes processing all the same batch of SR single data packets once. We aim to determine the optimal scheduling order, i.e., the time at which all embedded VNFs on each NFV node start supporting execution packet processing, each VNF scheduling cycle is repeated to reduce the total packet processing delay and the idle energy consumption of NFV nodes by following the optimal scheduling order.

We explain the scheduling process in discrete time, each scheduling period is divided into T time slots of equal length [11], VNF scheduling decisions are made at each time slot, when the binary decision variable $x_{ij}^t = 1$, it means that the data packets that need to be processed by a certain VNFs in the network service start to be processed at time slot t, Otherwise $x_{ij}^t = 0$. Different from only considering the makespan scheduling decision, we additionally introduce an continuous variable t_i to be used for the calculation of IEL, which indicates the end time of the last packet processing on the NFV node n_i within a scheduling period.

The important mathematical symbols and decision variables are shown in Table 1 and the discrete time MILP formulation of the energy-delay-aware VNF scheduling problem is presented as follows:

Objective

$$\min_{x_{ij}^t} C(M, E) \tag{1}$$

where

$$E = \sum_{i=1}^{|N|} [t_i - d_i] \cdot e_i \tag{2}$$

$$C(M, E) = \sigma M + (1 - \sigma)E \tag{3}$$

Constraints

$$\sum_{t=0}^{T} x_{ij}^t = 1, \forall j \in S, \forall i \in N_j \tag{4}$$

$$x_{ij}^t \cdot (t + p_{ji}) \leq t_i, \forall i \in N, \forall j \in S, \forall t \in T \tag{5}$$

$$\sum_{t=0}^{T} \left(t + p_{ji}\right) \cdot x_{ij}^t \leq M, \forall j \in S, \forall i \in N \tag{6}$$

$$\sum_{j \in S} \sum_{t' \in T_{ij}^t} x_{ij}^{t'} \leq 1, \forall t \in T, \forall i \in N, where \quad T_{ij}^t = \{t - p_{ji} + 1, \ldots, t\} \tag{7}$$

$$\sum_{t=0}^{T} \left(t + p_{j,\sigma_{h-1}^j}\right) \cdot x_{\sigma_{h-1}^j j}^t \leq \sum_{t=0}^{T} t \cdot x_{\sigma_h^j j}^t, \forall j \in S, \forall h \in \{1, 2, \ldots, L_{S_j} - 1\} \tag{8}$$

We combine the two objective of IEL and delay in the objective function, as shown in (3). The parameter σ can be adjusted to achieve any desired energy/delay tradeoff, a smaller σ will allow the model stresses energy loss minimization, while a larger σ emphasizes more delay optimization. Where (2) guarantees that the total IEL in a scheduling period is the sum of the IEL of all NFV nodes; Constraint (4) ensures that the traffic data packets of each SR are only processed once by the VNFs regard to specific NFV nodes in a scheduling period; Constraint (5) indicates the end time of the NFV node processing the data packet within a scheduling cycle; Constraint (6) indicates makespan to be the maximum value of the time for the last packet to finish processing on all NFV nodes; Constraint (7) indicates that each NFV node can only perform packet processing by one VNFs at a time; Constraint (8) indicates that the traffic processing of each SR must be performed in a specific order of the SFC, that is the VNFs priority constraint; scheduling objective (1) is to minimize the

Table 1 Summary of important notations and decision variables

Notation	Description
S	Set of network services:$\{1, 2, \ldots, \lvert S\rvert\}$
S_j	jth network service
T	A big enough number
L_{S_j}	The VNFs number of S_j
M	Makespan
N	Set of NFV nodes:$\{1, 2, \ldots, \lvert N\rvert\}$
N_j	Set of NFV nodes selected by VNFs of the j th network service
n_k	kth NFV node
f_{ij}	jth VNF of S_i
p_{ji}	Packet batch processing time of S_j at n_i
σ_h^j	NFV node index selected by the h th VNF of the network service S_j
E	Continuous variable indicating IEL of all NFV nodes in a VNF scheduling period
d_i	The total processing delay for n_i to process packets within a VNF scheduling period
e_i	Energy consumption per unit time slot of n_i
t_i	Continuous variable indicating the completion time of last packet on n_i

Notation decision variable	Description
x_{ij}^t	Binary decision variable indicating whether or not S_j starts being processed at the t th time slot on n_i

weighted sum of delay and IEL under the premise of satisfying the above-mentioned related VNFs execution constraints.

4 VNF scheduling scheme based on Q-learning

A Markov decision process is a five-tuple, (S, A, P_a, R_a, γ), where S is the state set, A is the action set, P_a is the transition probability to the next state after taking an action in specific state, R_a is the immediate reward after taking the action, γ is the discount factor ($0 \leq \gamma \leq 1$, and usually close to 1). Reformulate the discrete time model of VNF scheduling in the previous section as an MDP problem, we aim to use RL to solve this problem. For the VNF scheduling problem, the first focus is on the state and the specific action in the state, and there must be a unique correspondence between the state and the action, followed by the state transition mechanism, and finally the reward feedback mechanism of the learning system. According to literature [14], we design a new state representation and state transition rules. In addition, we also introduce a hierarchical reward enhancement mechanism and a new weighted reward function. The following is a detailed introduction of the algorithm.

4.1 System state and transition rules

The system state is perceived at the beginning of each time slot $t \in [0, T]$, which is defined as $s(t) = [M(t), F(t)]$. $M(t) = [m_1(t), m_2(t), \ldots, m_{|N|}(t)]$ indicates the state of the NFV node, $F(t) = [f_{1j}, f_{2j}, \ldots, f_{|S|j}]$ indicates the next VNFs of each SFC waiting to process packets, j indicates the jth VNFs of the SFC to which it belongs. We number each VNF according to the total length of all SFCs and the length of the SFC to which the VNFs belongs. Furthermore, if we use L to indicate the total number of VNFs owned by all SFCs. Then there is a one-to-one mapping between $f_{ij}, i \in S$ and the set of integers $0, 1, \ldots, L$, then you can use $f_l, l \in 0, 1, \ldots, L$ to represent each $f_{ij}, i \in S$. The system state is updated at the beginning of each time slot $t \in [0, T]$ until it reaches the last state of a scheduling cycle. At each time slot t, $m_k(t), k \in N$ is defined as follows:

$$m_k(t) = \begin{cases} 0, & \text{if } n_k \text{ is not processing packets} \\ 1, & \text{if } n_k \text{ is processing packets} \end{cases} \tag{9}$$

System transitions include state of NFV nodes $M(t)$ and waiting VNFs $F(t)$. The state transition rules of $M(t)$ are as follows:

$$m_k(t+1) = \begin{cases} 0, & \text{if } a_k(t) = 0, m_k(t) = 1, \theta_k(t) = 1 \\ 1, & \text{if } a_k(t) \neq 0, m_k(t) = 0, \theta_k(t) > 1 \\ m_k(t), & \text{otherwise} \end{cases} \tag{10}$$

where $a_k(t)$ refers to the VNFs selected to perform packet processing at the beginning of time slot t by n_k, the processing of VNFs is skipped if $a_k(t)$ equals $\{\phi\}$, node remains idle for a slot of time. Otherwise, the processing of a certain VNF is required. $\theta_k(t)$ indicates the remaining packet process time on NFV node n_k at time t.

The state transition rules of $F(t)$ are as follows:

$$F_i(t+1) = \begin{cases} F_i(t) + 1, & \text{if } \tau_{S^i_{!tail}}(t) = 1 \\ -1, & \text{if } \tau_{S^i_{tail}}(t) = 1 \\ F_i(t), & \text{otherwise} \end{cases} \tag{11}$$

where $S^i_{!tail}$ is a marker representing the non-last VNFs of S_i, while $\tau_{S^i_{!tail}}(t)$ denotes the remaining packet batch processing time for the VNFs at time slot t, similar to S^i_{tail} and $\tau_{S^i_{tail}}(t)$. Equation (11) indicates that if a certain non-last VNFs for SFC perform packet processing, then use the next VNFs waiting to process data packets of the SFC to which the VNFs belong to update state; if the last VNFs data packet of a certain SFC is processed, then use -1 to update.

In the initial state, all NFV nodes are idle, no VNFs perform operations, and all VNFs embedded on NFV nodes waiting to perform packet processing. Use s_{ini} and s_{ter} to represent the initial state and termination state of a VNF scheduling cycle respectively, according to the system state and transition rules, we have $s_{ini} = (0, \ldots, 0, 0, \ldots, \mathrm{L} - L_{S_{|S|}})$ and $s_{ter} = (0, \ldots, 0, -1, \ldots, -1)$.

4.2 Action set and reward function

At the beginning of each time slot in each scheduling cycle, the scheduler executes actions consisting of the next VNFs waiting to perform data processing on each NFV node. Denote the action taken by the VNF scheduler at t slot time with $A(t)$, then $A(t) = (3, 5, 6, 8)$ indicated f_3, f_5, f_6, f_8 on N_1, N_2, N_3, N_4 will perform packet processing, respectively. We denote the state at time slot t by $s(t)$, the cartesian product $A(s_t) = A_1(s_t) \otimes A_2(s_t) \otimes \cdots \otimes A_{|N|}(s_t)$ in the algorithm to represent the feasible actions in each state, where $A_k(s_t)$ denotes the index of VNFs that can perform packet processing of the NFV node N_k in state $s(t)$. The number of feasible action sets for each state is not the same, sometimes there is more than one action, and sometimes there is only one feasible action. In addition, if a node in a certain state has no VNFs capable of performing packet processing, then $A_k(s_t) = \{\phi\}$, this has the same meaning as the $a_k(t)$ we mentioned in the previous subsection.

The set of possible actions in each of the above states depends only on the state, regardless of other factors, that is,there is a one-to-one mapping relationship between $s(t)$ and $A(s_t)$, and the feasible action set can be known through $s(t)$. The algorithm for finding feasible action sets in each state in our experiment is similar to that in literature [14], only the algorithm is adjusted according to the newly designed system state and state transition rules.

The objective of this paper is to implement VNF scheduling that is aware of energy consumption and delay, both delay and energy consumption are relative to a scheduling cycle and these two indicators can only be obtained after a scheduling cycle. Therefore, it is different from the instant reward in the training process of the classic RL algorithm(i.e. get a reward feedback immediately after taking an action). We use the cumulative reward method to measure the merits of a series of actions taken, according to the obtained makespan and IEL after a whole scheduling period, the cumulative reward is calculated using the formula (12).

$$R(s_t, A_t) = \omega c_t / M + (1 - \omega) c_e / E, \forall (s_t, A_t) \in \Omega \tag{12}$$

Where M and E are the makespan and IEL obtained from a round of learning and training respectively. The smaller M or the smaller E, the greater the reward. Ω contains state-action pairs with the number of actions ≥ 2 in all feasible action sets during the current episode of exploration. c_t and c_e are the weight coefficients for adjusting the rewards for minimizing makespan and minimizing IEL. ω is used to solve the scheduling scheme for different goals, only need to minimize makespan or minimize IEL, then set it to 1 or 0 respectively, when we use both makespan and IEL as scheduling goals, it needs to be a number between 0 and 1.

Algorithm 1 Q-learning algorithm using hierarchical reward enhancement

Initialization:
Initialize $n = 0, b = 1, b_{step}, const_{value}, n_{max}, \alpha, \epsilon, \gamma$;
while $n < n_{max}$ **do**
  Set $s_t \leftarrow s_{ini}$;
  Set $\Omega_n \leftarrow \emptyset$;
  while $s_t \neq s_{ter}$ **do**
    $A_t \leftarrow$ select action according $\epsilon-$greedy policy;
    if number of cartesian action at $s_t \geq 2$ **then**
      Add (s_t, A_t) to Ω_n;
    end if
    take A_t action at current state, state transfer to s_{t+1};
    while True **do**
      if $s_{t+1} = s_{ter}$ **then**
        break;
      end if
      $A(s_{t+1}) \leftarrow$ get cartesian action set at s_{t+1};
      if $A(s_{t+1}) \neq A_{null}$ **then**
        break
      else
        take A_{null} action at s_{t+1};
      end if
    end while
  end while
  $n \leftarrow n + 1$
  Calculate $R(s_t, A_t)$ according to (11), add $R(s_t, A_t)$ to $R(s_t, A_t)_{list}$;
  for all $(s_t, A_t) \in \Omega_n$ **do**
    if $R(s_t, A_t) \geq \max(R(s_t, A_t)_{list})$ **then**
      $R(s_t, A_t) \leftarrow R(s_t, A_t) * \min(const_{value}, b)$;
      $b \leftarrow b + b_{step}$;
    end if
    $Q(s_t, A_t) \leftarrow (1-\alpha)Q(s_t, A_t) + \alpha(R(s_t, A_t) + \gamma \max_{A_{t+\Delta t}} Q(s_{t+\Delta t}, A_{t+\Delta t}))$;
    Add $R(s_t, A_t)$ to $R(s_t, A_t)_{list}$;
  end for
end while

4.3 RL-HRE algorithm

The RL-HRE algorithm is designed based on the Q-learning algorithm, which can learn the desired VNF scheduling strategy through continuous interaction with the system environment [24, 25]. Since the feasible action of each system state are obtained from the various action set using the $\epsilon-$greedy algorithm [26], and different actions often make the span of time slots between two Q value update states different in the RL-HRE learning process, therefore use the following formula 13 to update the Q value in the Q-table.

$$Q(s_t, A_t) = (1-\alpha)Q(s_t, A_t) + \alpha[R(s_t, A_t) + \gamma \max Q(s_{t+\Delta t}, A_{t+\Delta t})] \tag{13}$$

where $Q(s_t, A_t)$ indicates the Q value taken under the state s_t in the Q-table, which can be regarded as the probability of taking in the state, and the larger the Q value, the greater the probability that the scheduling decision will choose this action. α is the learning rate, γ is the discounter factor, both are the numbers between 0 and 1, α control the Q value update step size, α is the closer to 0, the more the Q value changes during the update process; γ quantifies how much importance we give for future rewards, γ is closer to 0, the learning agent will consider future rewards with greater weight. Δt reflects the unequal length of time slots between states when updating the Q value caused by executing different actions. Assuming that after enough episodes of

training of the algorithm, the Q table converges to the optimum, denoted by Q^*, we use the ϵ greedy algorithm to make the best VNF scheduling strategy according to Q^*, as shown in the formula 14.

$$\pi^* = arg\max Q^*(s_t, A_t) \tag{14}$$

The main pseudocode of RL-HRE is shown in Algorithm 1, where n and n_{max} represent episode counter and the maximum number of training episodes, line 7 contains the algorithm to find the feasible action set in a specific state and select the action through the Q table according to the greedy algorithm; lines 8–10 indicate that in the n episode of training, all corresponding state and action pairs that conform to the Cartesian product feasible action set with the number of actions greater than or equal to two need to be added to Ω_n, if the number of actions equals to one is added to the Ω_n, unnecessary updates will be performed when the Q value is updated after an episode of action exploration. When a cycle ends, we calculate the cumulative rewards obtained from a series of scheduling decisions taken in this episode according to line 25 and add the cumulative reward to a cumulative reward list. During the Q-table update process, if the cumulative reward is found to be greater than or equal to the maximum value in the cumulative reward list so far, update the cumulative reward used for this episode of Q value update through line 28, where b is a constant with an initial value of one, b_{step} is the step size of each increment, and the $const_{value}$ is a constant used to control the upper bound of the reward. This is the introduced hierarchical reward enhancement mechanism, during the entire training process, whenever a better scheduling scheme is encountered, the cumulative reward for the Q value update will be updated to a larger and larger one with an upper bound value. The overall framework of the algorithm is shown in Fig. 3, the cumulative reward obtained after an episode of training will through one of two operations(i.e. formula 12 scheduling plan reward calculation and HRE) before being used to update the Q value, and finally the cumulative reward is used to update the Q value.

5 Simulation and performance evaluation

In the VNF scheduling simulation experiment, VNFs are randomly embedded on NFV nodes, the number of VNFs requested by each network service is set as a random integer between 2 and 5, and the packet batch processing time of a VNFs is set as an integer between 1 and 5, energy consumption per unit time of NFV nodes is set as an integer between 1 and 10. Instances of VNF scheduling problems under different network scales (the number of SR and the number of NFV nodes are different) are pre-defined and remain unchanged during the algorithm training process. The problem instances are obtained by random sampling according to the above parameter settings. For the MILP formula, we use the Gurobi optimization solver to find the optimal solution, and use Python to model and simulate the scheduling algorithm. All experiments are performed on an Intel i5-7200U CPU@2.5GHz.

5.1 Convergence comparison of RL-HRE and RL-Primitive

Under the medium-scale scheduling network instance ($N = 8, S = 10$), we compare the convergence of RL-HRE and RL-Primitive. In order to optimize for different purposes in RL-Primitive, we use the weighted reward function proposed in this paper. The two algorithms obtained experimental results under the best performance by adjusting the experimental parameters. The c_t and c_e in the reward function are set to 600 and 1400, α and ϵ are decay numbers (decaying from 1.0 to 0.01 with a decay rate of 0.998 and from 0.8 to 0.1 with a decay rate 0.995), b_{step} and $const_{value}$ are set to 2 and 6 respectively. As shown in Fig. 4, both algorithms showed convergence after about 1000 training epochs. For the goal of minimizing time delay only is denoted by M, the convergence curves of the two algorithms are almost the same, and for both energy-delay minimization or energy minimization only, the performance of our proposed RL-HRE is better than that of RL-Primitive. Through solve optimal solution by Gurobi solver, we also found that RL-HRE almost converged to the optimal solution, while RL-Primitive fell into a local optimum, this is because different cumulative rewards resulting from different solutions about energy minimization problems are more, the optimality of the solution is measured by the cumulative reward. Due to the ϵ-greedy algorithm, the algorithm has the same probability of obtaining different solutions in the early stage of training, which can easily lead to more suboptimal solutions than the optimal solution encountered before convergence during the algorithm exploration process. Without HRE, the reward feedback mechanism continuously enhances the Q value of the suboptimal solution, eventually causing the algorithm to fall into a local optimal solution. The introduction of HRE differentiates the optimal solution Q value and suboptimal solution Q value strengthening mechanism (i.e., the optimal solution that the agent encounters for the first time during exploration may receive a greater reward than the sum of the rewards obtained by encountering multiple sub-best solutions.). The convergence curve shows that the performance of our proposed algorithm is better than RL-Primitive.

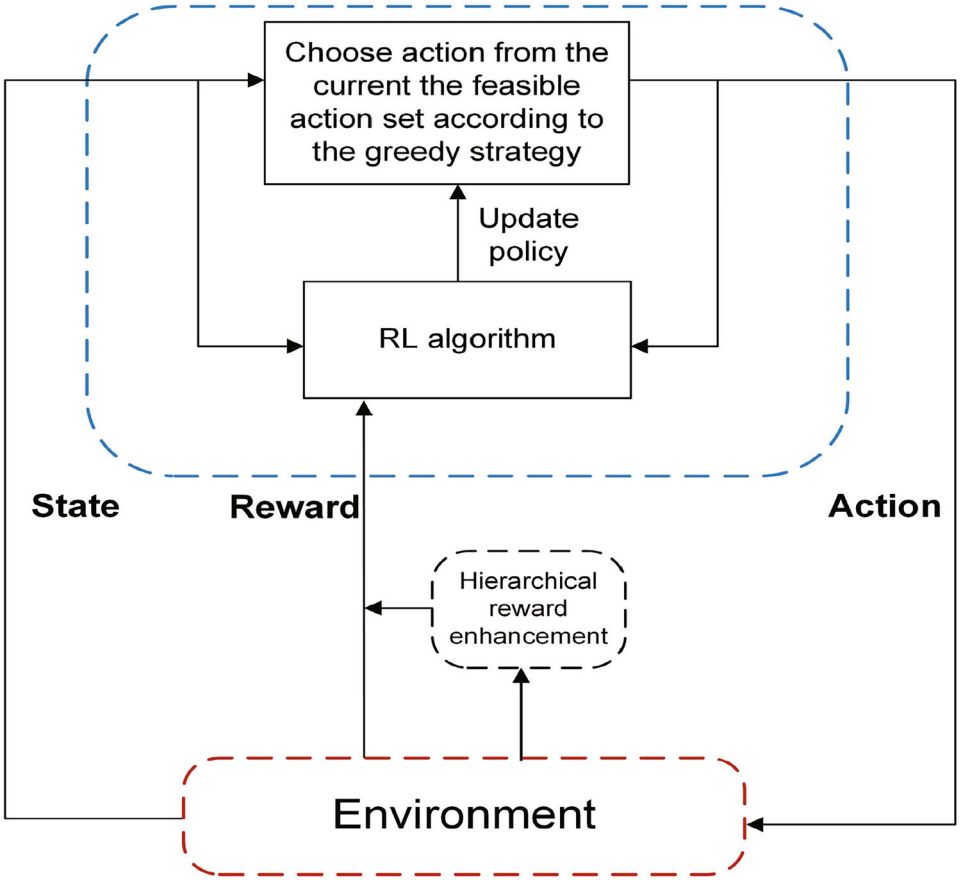


Fig. 3 Reinforcement learning framework using HRE

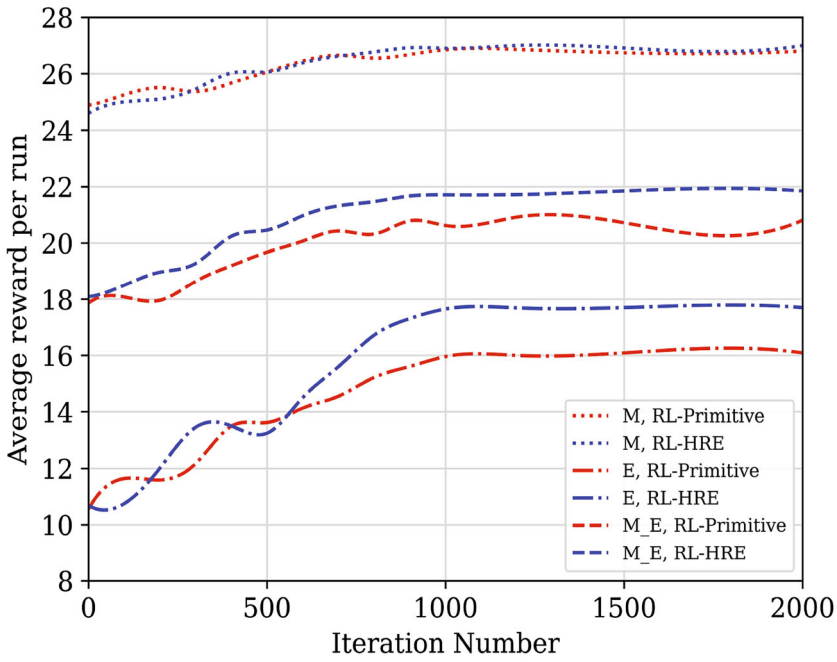


Fig. 4 Convergence of the RL-HRE compare to RL-Primitive

5.2 Energy delay controllable VNF scheduling

In this section, we verify the flexible energy-delay-aware scheduling of RL-HRE on a medium-scale scheduling network instance ($N = 8, S = 10$). We consider three network scenarios in real scenario: 1. Some NFV nodes (i.e. network devices) are old relatively, the energy consumption per unit time slot is relatively high, and the delay requirements of customers may not be so strict, then in order to minimize the energy loss as the goal and ignore the devices delay, the w value of the weighted reward function is set to 0; 2. network services have high requirements on latency regardless of the energy consumption of network devices; 3. not only does the latency have low requirements, but also the energy consumption of the device also needs to be as low as possible, similar to the first scenario, we set w to 1 and 0.5 for the latter two scenarios respectively. The rest of the parameters of the algorithm are set as the optimal parameters, run the RL-HRE algorithm 20 times for each scenario, and average the resulting scheduling delay and IEL every 4 times.

The above VNF scheduling results are shown in Fig. 5. The x-axis is the ratio of the average delay of the scheduling scheme obtained by the RL-HRE algorithm to the optimal delay of the optimal scheduling scheme obtained by Gurobi, the y-axis is the ratio of average IEL to optimal IEL and three different signs represent different scenarios. From the figure we can see that the average delay of the scheduling scheme obtained for the purpose of minimizing the delay is close to the optimal value as for makespan index, but the IEL index is far from the optimal value; the average IEL of the scheduling scheme with the goal of minimizing IEL is close to the optimal value as for IEL index, but the delay index is worse than the result of previous scenario; Compared with the previous two scenarios, the two indicators in the $w = 0.5$ scenario have more balanced performance.

The detail of the two specific VNF scheduling schemes in the two scenarios obtained in the above experiment are shown in Fig. 6, the same color block indicates VNFs belonging to the same network service, indicated by f_{ij} in the upper left corner of the color block. The scheduling scheme (a) is obtained in the experimental scenario of $w = 1$ and the scheduling scheme (b) is obtained in the $w = 0.5$ scenario. According to the optimal scheduling scheme obtained by using Gurobi, it can be seen that the delay of (a) reaches the optimal value but the IEL does not, and the delay and IEL of (b) both reach the optimal value. This demonstrates that our algorithm achieves a flexible energy-delay-aware VNF scheduling scheme solution through a weighted reward function.

5.3 Comparison with related algorithms

To verify the good solution accuracy and low computational complexity of RL-HRE, we compare the average reward and running time of RL-HRE for solving the VNF scheduling problem of minimizing IEL and delay with some other methods. For the calculation of the average reward, all methods are calculated using the same weighted reward function 12 proposed in this paper, which can

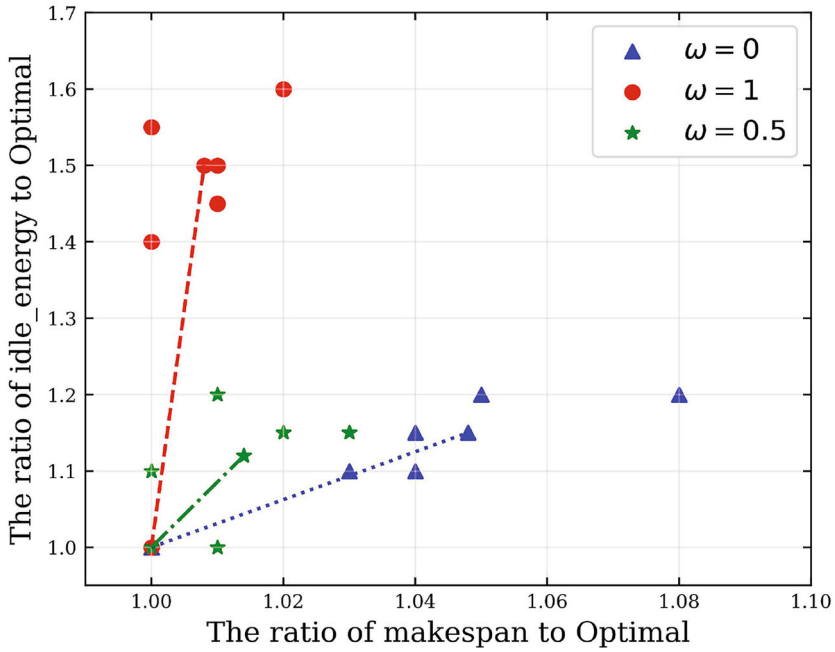


Fig. 5 Energy-delay controllable scheduling using the RL-HRE

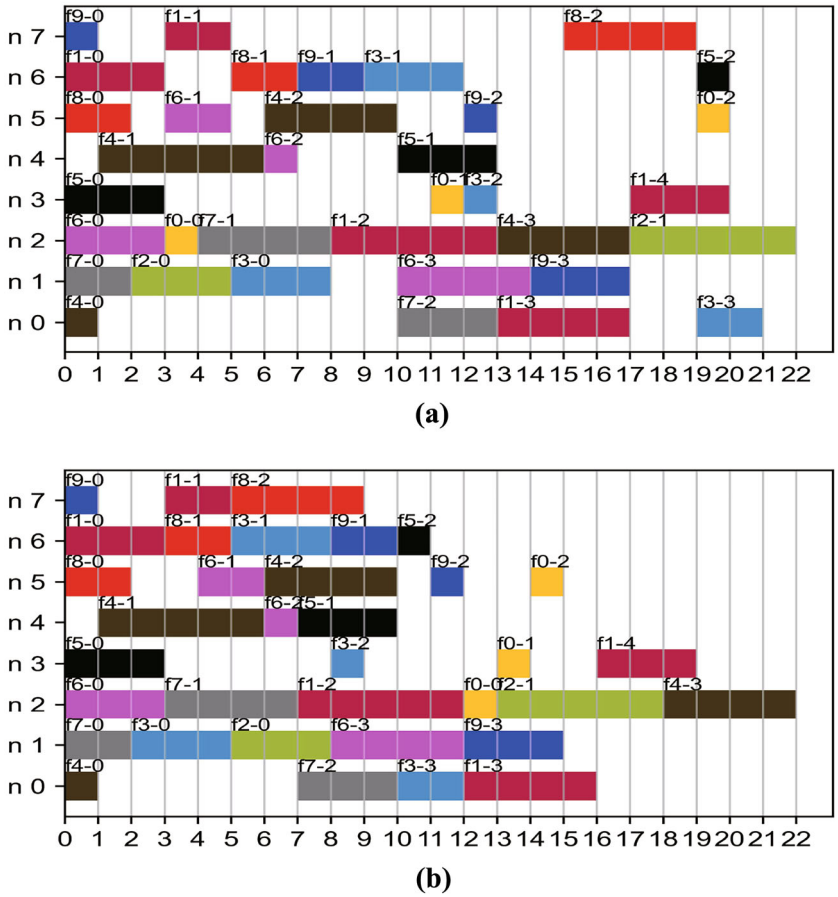


Fig. 6 Energy-delay-aware verification of RL-HRE

effectively evaluate the performance of scheduling scheme delay and IEL index.

The comparison of the results of the average reward and average running time of each algorithm is shown in Fig. 7, Genetic algorithm (GA) [9, 27] is a classic meta-heuristic algorithm for JSP solving, Greedy fast processing (GFP) always chooses the VNF with the shortest processing time each time. (a) shows the running results of medium-scale scheduling network instances. The running time of RL-Primitive and RL-HRE is almost three times faster than MILP, but in terms of solution accuracy, the RL-HRE we mentioned is closer to the optimal value than RL-Primitive. Although the solution accuracy of the GA algorithm is comparable to that of RL-Primitive, its running time is even worse than that of MILP. The running time of random scheduling and GFP is almost negligible, but the average reward of the solution they compute is much smaller than the optimal value. Similar to (a), (b) is the running results of a large-scale scheduling network instance($N = 10, S = 15$), as the network size increases, the running time of MILP and GA increases a lot. The running time of RL-HRE is almost 4 times smaller than that of MILP, but its solution accuracy is still close to the optimal value, and it is better than RL-Primitive, this is because, in the process of updating the Q value of the RL-HRE algorithm, the case that the Cartesian product has only one feasible action is omitted, thereby reducing the computational complexity, which is more obvious in large-scale scheduling network instances.

6 Conclusions and future work

This paper considers the energy loss of NFV nodes in the VNF scheduling process under the SDN/NFV network architecture, and proposes a MILP model with the goal of minimizing delay and idle energy loss, a new Q-learning algorithm called RL-HRE using a hierarchical reward enhancement mechanism is designed to solve this problem, that can better realize the situation where the agent falls into the local optimal solution due to randomness in the early exploration stage while ensuring stability, and flexible energy-delay-aware VNF scheduling is realized through a weighted reward function. Finally, the simulation results show that RL-HRE has a better solution accuracy than RL-Primitive in solving the energy-minimizing objective scheduling scheme, and is superior to other

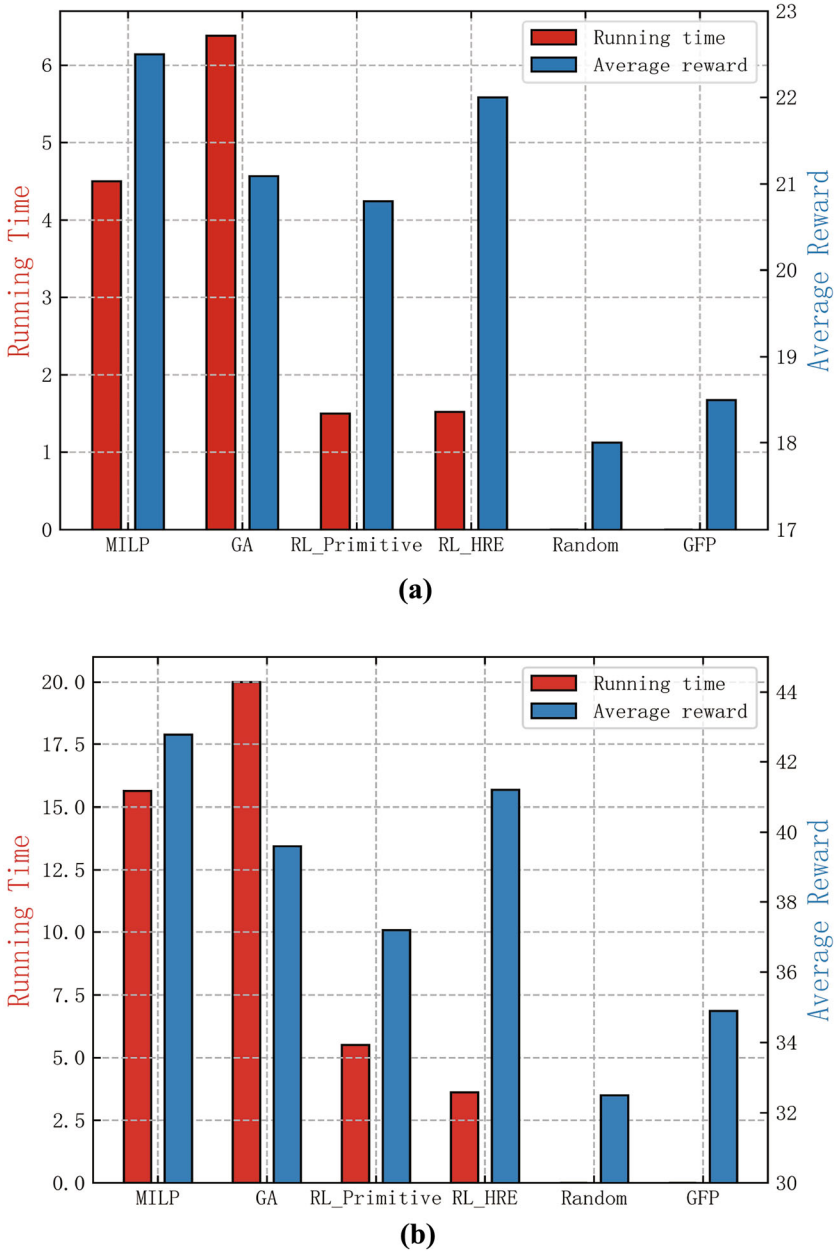


Fig. 7 Comparison of running time and average reward

comparison algorithms in terms of solution accuracy and computational complexity.

The algorithm presented in this paper demonstrates improved performance exclusively on a specific problem instance. To further enhance the applicability of the trained model to unknown problem instances of a similar nature, future research should explore the development of a more comprehensive reinforcement learning algorithm that learns from a batch of problem instances. Additionally, given the diverse nature of energy consumption modeling in job-shop scheduling problems, it is recommended that a future algorithm be designed to accommodate multiple energy loss models.

Author contributions XWY: conceptualization, software, writing-original draft. XCW:supervision, writing-reviewing, methodology. YS:writing-reviewing, editing. GT:writing-reviewing, editing.

Funding The work of this paper was supported by the Zhejiang Province Natural Science Foundation (No. LY19F020002, No. Y19F020031), and the Zhejiang Provincial Key Laboratory of New Network Standards and Application Technology (No. 2013E10012).

Data availability No datasets were generated or analysed during the current study.

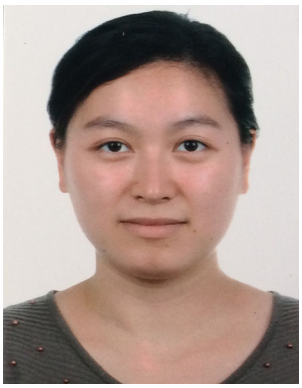
Declarations

Competing interests The authors declare no competing interests.

References

1. Ordonez-Lucena, J., Ameigeiras, P., Lopez, D., Ramos-Munoz, J.J., Lorca, J., Folgueira, J.: Network slicing for 5g with SDN/NFV: concepts, architectures, and challenges. IEEE Commun. Mag. **55**(5), 80–87 (2017)
2. Yousaf, F.Z., Bredel, M., Schaller, S., Schneider, F.: NFV and SDN-key technology enablers for 5g networks. IEEE J. Sel. Areas Commun. **35**(11), 2468–2478 (2017)
3. Herrera, J.G., Botero, J.F.: Resource allocation in NFV: a comprehensive survey. IEEE Trans. Netw. Serv. Manag. **13**(3), 518–532 (2016)
4. Riera, J.F., Escalona, E., Batalle, J., Grasa, E., Garcia-Espin, J.A.: Virtual network function scheduling: concept and challenges. In: 2014 International Conference on Smart Communications in Network Technologies (SaCoNeT), pp. 1–5. IEEE (2014)
5. Riera, J.F., Hesselbach, X., Escalona, E., García-Espín, J.A., Grasa, E.: On the complex scheduling formulation of virtual network functions over optical networks. In: 2014 16th International Conference on Transparent Optical Networks (ICTON), pp. 1–5. IEEE (2014)
6. Garey, M.R., Johnson, D.S., Sethi, R.: The complexity of flowshop and jobshop scheduling. Math. Oper. Res. **1**(2), 117–129 (1976)
7. Brucker, P., Knust, S.: Complex Scheduling. GOR-Publications. Springer, Berlin (2006)
8. Tang, Xiongyan, Cao, Chang, Wang, Youxiang, Zhang, Shuai, Liu, Ying, Li, Mingxuan, He, Tao: Computing power network: the architecture of convergence of computing and networking towards 6g requirement. China Commun. **18**(2), 175–185 (2021)
9. Pezzella, Ferdinando, Morganti, Gianluca, Ciaschetti, Giampiero: A genetic algorithm for the flexible job-shop scheduling problem. Comput. Oper. Res. **35**(10), 3202–3212 (2008)
10. Long, Qu., Assi, Chadi, Shaban, Khaled: Delay-aware scheduling and resource optimization with network function virtualization. IEEE Trans. Commun. **64**(9), 3746–3758 (2016)
11. Alameddine, H.A., Qu, L., Assi, C.: Scheduling service function chains for ultra-low latency network services. In: 2017 13th International Conference on Network and Service Management (CNSM), pp. 1–9. IEEE (2017)
12. 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 (2017)
13. Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S.: Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv:1611.09940, (2016)
14. Li, Junling, Shi, Weisen, Zhang, Ning, Shen, Xuemin: Delay-aware VNF scheduling: a reinforcement learning approach with variable action set. IEEE Trans. Cognitive Commun. Netw. **7**(1), 304–318 (2020)
15. Mijumbi, R., Serrat, J., Gorricho, J.-L., Bouten, N., De Turck, F., Davy, S.: Design and evaluation of algorithms for mapping and scheduling of virtual network functions. In: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), pp. 1–9. IEEE (2015)
16. Promwongsa, N., Ebrahimzadeh, A., Glitho, R.H., Crespi, N.: Joint VNF placement and scheduling for latency-sensitive services. IEEE Trans. Netw. Sci. Eng. **9**(4), 2432–2449 (2022)
17. Thanh, N.H., Kien, N.T., Van Hoa, N., Huong, T.T., Wamser, F., Hossfeld, T.: Energy-aware service function chain embedding in edge-cloud environments for IoT applications. IEEE Internet Things J. **8**(17), 13465–13486 (2021)
18. Soualah, O., Mechtri, M., Ghribi, C., Zeghlache, D.: Energy efficient algorithm for vnf placement and chaining. In: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pp. 579–588. IEEE (2017)
19. Duan, Jianguo, Wang, Jiahui: Energy-efficient scheduling for a flexible job shop with machine breakdowns considering machine idle time arrangement and machine speed level selection. Comput. Ind. Eng. **161**, 107677 (2021)
20. Maassen, K., Perez-Gonzalez, P., Günther, L.C.: Relationship between common objective functions, idle time and waiting time in permutation flow shop scheduling. Comput. Oper. Res. **121**, 104965 (2020)
21. Ye, Q., Zhuang, W., Li, X., Rao, J.: End-to-end delay modeling for embedded VNF chains in 5g core networks. IEEE Internet Things J. **6**(1), 692–704 (2018)
22. Gao, Kaizhou, Huang, Yun, Sadollah, Ali, Wang, Ling: A review of energy-efficient scheduling in intelligent production systems. Complex Intell. Syst. **6**, 237–249 (2020)
23. Ku, W.-Y., Beck, J.C.: Mixed integer programming models for job shop scheduling: a computational analysis. Comput. Oper. Res. **73**, 165–173 (2016)
24. Melo, F.S.: Convergence of q-learning: a simple proof. Institute Of Systems and Robotics, Tech. Rep, pp. 1–4 (2001)
25. Watkins, C.J.C.H., Dayan, P.: Q-learning. Mach. Learn. **8**, 279–292 (1992)
26. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, New York (2018)
27. Golberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning, p. 36. Addison Wesley, Boston (1989)

Xi-Wei Yang is pursuing a master's degree in Electronic Information from the School of Information and Electronic Engineering, Zhejiang Gongshang University. His main research interests include next generation network technology architecture, machine learning, etc.



Xiao-Chun Wu received the Ph.D. degree in Computer Science and Technology from Zhejiang university, China in 2013. She had one year visiting experience at the Department of Computer Science, Northwestern University, Evanston, IL, USA from 2017 to 2018. She is currently an associate professor with the School of Information and Electronic Engineering, Zhejiang Gongshang University, China. Her research interests include Forwarding and Control Element Separation (ForCES), Software-Defined Networking (SDN), Network Function Virtualization (NFV), and network security.

Yue Shao is pursuing a master's degree in Electronic Information from the School of Information and Electronic Engineering, Zhejiang Gongshang University. His main research interests include next generation network technology architecture, knowledge graph, machine learning, etc.

Gan Tang is pursuing a master's degree in Electronic Information from the School of Information and Electronic Engineering, Zhejiang Gongshang University. His main research interests include next generation network technology architecture, knowledge graph, machine learning, etc.