修士論文

Deep Q-Learning に基づくネットワークスライス資源割り当てアルゴリズム

Network Slicing Resource Allocation Algorithm Based on Deep Q-Learning

令和 4 年 1 月 27 日 提出

指導教員　　相田　仁 教授

電気系工学専攻

37－195107　　周　鴻霖

Master Thesis

Network Slicing Resource Allocation Algorithm
Based on Deep Q-Learning

Deep Q-Learning に基づくネットワークスライス
資源割り当てアルゴリズム

Submitted on January 27th, 2022

Supervisor Professor Hitoshi Aida

Department of Electrical Engineering and Information Systems

37−195107 Honglin Zhou

Abstract

5G networks are expected to support different vertical industries that are characterized by diverse performance requirements. Network slicing has been identified as the backbone of the rapidly evolving 5G technology which is considered to be the key enabler to enhance cellular networks with the desired flexibility to achieve this target. However, since slices share the wireless resources of the entire system, the allocation of resources between slices could be a key problem. Considering the dynamic nature of the network request, the load between the slices will also change, which results in excess or insufficient resources of the slice.

This thesis builds a wireless virtual network resource management system based on the architecture of software-defined network (SDN). The main purpose is to maximize the resource utilization of the entire system under the premise of ensuring quality of service (QoS) satisfaction. In order to dynamically adapt to changes in network requirements, this thesis proposes a resource allocation scheme based on deep Q-network (DQN) and evaluates it by simulating and comparing with other existing methods. As a result, the proposed can adjust the resource allocation to guarantee the user QoS requirements and maintain balance between slices comparing to traditional methods.

Contents

1 Introduction

The fifth-generation (5G) cellular networks have gone into the commercialization phase. 5G is touted as the generation of mobile networks that will support dedicated use-cases and provide specific types of services to satisfy various customer demands simultaneously. Which promise to cater for the telecommunications sector and "vertical industries" including autonomous driving, smart factories, the health sector, and so on. Differentiated business needs and application scenarios make 5G meet different network and functional requirements in terms of mobility, billing, security, policy control, delay and reliability [1, 2]. Three major application scenarios of 5G networks are considered by the Next Generation Mobile Network (NGMN) in "5G White Paper" as enhanced Mobile Broadband (eMBB), massive Machine Type Communication (mMTC) and ultra-high Reliability and Low Latency Communication (uRLLC) [3] as shown in Fig. 1.

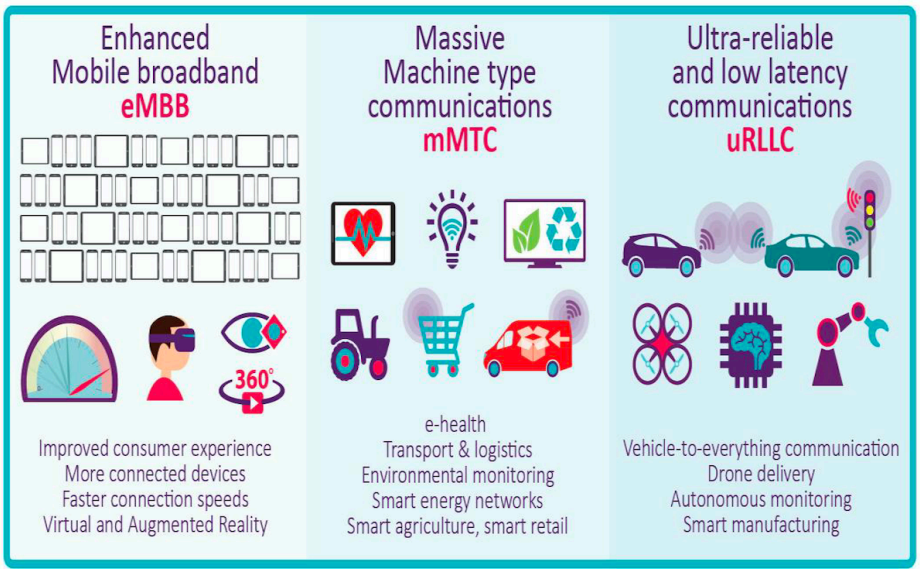


Fig.1: Service Scenarios of 5G [4]

The eMBB application scenario is mainly aimed at high-traffic mobile broadband services. The next-generation communication network needs to improve transmission capabilities, increase the speed of data upload and download, and promote the better realization of services with very large data size, such as the large-scale transmission of video data [5]. In the future, people expect faster networks to achieve entertainment, such as faster social communication, smooth ultra-high-definition video, amazing virtual reality (VR) or augmented reality (AR) technology, etc. In fact, this application scenario is for the communication needs of people's daily life.

The application scenario of uRLLC have been widely studied. The Internet of Vehicles (IoV) is a specific application of this scenario. On the one hand, IoV requires high reliability,

that is, the network has high stability and strong anti-interference ability to ensure that the data transmission of the car is always smooth and the data accuracy is high. On the other hand, since cars need to make quick decisions based on a large amount of environmental information, IoV requires extremely low latency to transmit data in real time [6]. In addition, industrial automatic control and remote surgery also require high-reliability and low-latency communication requirements to complete real-time operations.

The mMTC application market has huge potential. In the future, the field of Internet of Things (IoT) is a hot spot in the communication industries. Countless infrastructures will be connected to the network. Allowing various household appliances to connect to the network, the users can control them through the network to achieve smart homes. With the connection street lights, trash cans and other municipal facilities to IoT, city managers can easily to maintain the system [7]. The characteristics of IoT are mainly large connections and massive data. The future communication network needs to connect a large number of terminal devices. Therefore, the access capability of the network needs to be improved, and the large connection also makes the network more complicated. How to manage the large connection network is a big challenge [8]. In addition, a large number of connected terminals will generate massive amounts of data, which requires reliable big data processing and transmission capabilities.

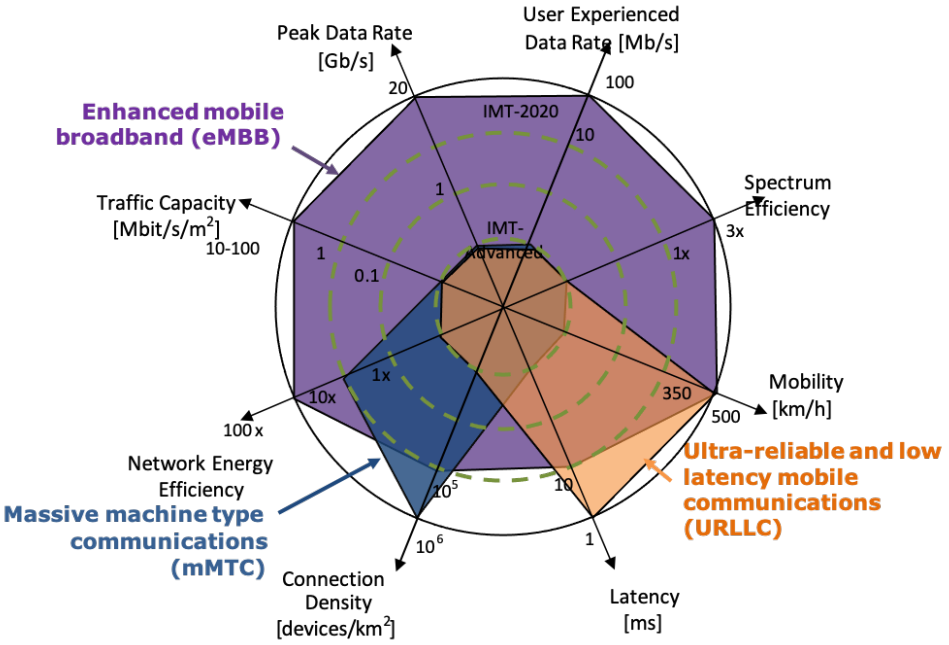


Fig.2: 5G Service Scenarios and Requirements [3, 4]

From Fig.2, it can be seen that the three application scenarios rely on different communication requirements, while traditional communication networks can only meet a single demand. Therefore, operators need to establish dedicated networks for the three types of application scenarios. This undoubtedly increases the operator' s cost, lowers the flexibility of the network, and it also slows down the widespread promotion of the new network. Although

5G promises to extend the performance range, provide higher data rates, lower latency and higher connection density, most services do not require extreme configuration of all these performance metrics. Instead, the network should build network slices in a flexible and efficient manner to meet the capacity and coverage requirements of specific services [9].

In order to support various business scenarios with different performance requirements on one physical network, network slicing (NS) [10] is considered as one of the most promising architecture in the upcoming 5G. Through network slicing technology, operators can divide physical infrastructure into multiple virtual networks according to the needs of different users to meet communication in various scenarios, and flexibly establish various slice networks to provide different services according to requirements, and realize communication in multiple application scenarios.

One of the issues that wireless networks need to pay attention to is that with the increase of end-user traffic, wireless spectrum resources will become scarce. This problem can be solved by considering new access technologies to improve wireless network efficiency, such as heterogeneous networks, a combination of different radio access technologies (RAT), and the use of differentiated services or cognitive radio. However, these solutions might also increase the cost of network operators, thereby making network management and operation more complex, and therefore requiring the deployment of more infrastructure.

To reduce the operating costs of network service providers, wireless network virtualization (WNV) technology is introduced to help the implementation of network slicing. WNV is the combination or division of a set of network resources and presents (abstracts) it to users, so that each user has a separate network view. Resources can be basic (nodes, links) or derived (topology), and can be virtualized recursively [11]. In short, the goal of network virtualization is to create logical partitions of some existing physical network resources in an effective way. Aiming at wireless network scenarios where wireless resources are shared between slices, this article conducts related research on resource allocation and isolation in wireless networks.

The rest of this thesis will be organized in following fashion. The next chapter will introduce the details of network slicing resource allocation and related works in these years. It will also introduce the basic concept of related theories and technologies, including deep learning, deep reinforcement learning and software defined network technologies. The third chapter designs the wireless resource management system between network slices and formulates the optimization problem. The proposed method to solve the problem is explained in chapter 4. Chapter 5 discusses the simulation results and evaluates the method by comparing the performance with other two methods. Finally, the last chapter consists of conclusion about the overall thesis and the potential future works.

2 Related Work

2.1 Resource Allocation

The challenge of realizing wireless network slicing mainly lies in changes in wireless link capacity and resource constraints. In wireless communication, the maximum transmission rate that a user can obtain depends on the signal-to-noise ratio (SINR) and the available bandwidth of the link. In addition, the distance between the transmitter and the receiver, the location of the interference source or the obstacles between the communication nodes, the SINR will also change accordingly. Compared with wired network deployment, the available radio spectrum is regulated and restricted, so it cannot be easily increased. Therefore, for slices on the RAN side, the problem that needs to be considered is how to allocate appropriate resources for the slices. The slicing problem in the cellular network environment mainly includes the following three types of methods:

(1) PRB scheduling: Y. Zaki et. al [12] proposed a virtualized LTE [13] base station, which is, eNodeB (eNB) architecture to allow different operators to share the same physical resources. The solution is based on a virtualized controller module (such as virtualized CPU), which hosts different virtual nodes, allocates resources, and is responsible for spectrum sharing and data reuse. The controller module will complete two tasks: connecting multiple virtual eNBs to the physical eNB and allocating the physical resources in it. It will also allocate the wireless resources between different virtual eNBs. For the second task, the solution uses physical resource blocks (PRB) as the smallest resource granularity that can be allocated and allocates them to different virtual nodes instead of user equipment. According to the pre-signed contract, the PRB will be allocated to different virtual eNBs.

Literature [14] proposed a Karnaugh map embedding algorithm (KEA) to deal with the virtual wireless network (slice) request to embed physical wireless resources. Specifically, this work focuses on allocating resources to slices when a request comes (online request). Compared with the resource allocation (offline request) where all requests arrive at once, this method has some obvious disadvantages. In order to deal with this dynamic scene, requests are grouped in a time window, and embedded at the end of each window to complete. The spectrum is modeled as a two-dimensional (frequency and time) grid of allocatable resources and is allocated based on the Karnaugh map algorithm. The author does not explain how to implement this mechanism in real hardware or current wireless technology.

(2) Slice scheduling: Network Virtualization Substrate (NVS) [15] proposed an architecture and algorithm for WiMAX network slicing. The goal is to meet the resource requirements of slicing, so that the slices can avoid causing interference with each

other. In this case, scheduling is achieved by modifying the WiMAX stream scheduler at a higher level rather than the PRB scheduler. The scheduler will select a slice in each scheduling frame to allocate resources for it. Then, the slice will schedule the data packet according to its corresponding algorithm, and finally send the data packet to the frame scheduler. Therefore, the frame scheduler has not changed in the way PRBs are allocated, but this needs to be modified at the MAC layer of the base station. This leads to this approach facing deployment constraints similar to PRB scheduling recommendations, because in general, the software is proprietary and manufacturer-dependent.

(3) Traffic shaping: Virtual base station [16] is a virtualized architecture of WiMAX base station (BTS), which is used to realize resource sharing and isolation between multiple virtual network slices. This solution adds a new layer called a virtual BTS substrate to the WiMAX network. The substrate acts as a virtualization layer and provides a platform for each slice to create and execute a virtual machine (VM). The framework contains two important aspects: the definition of a virtual BTS as an entity separated from the physical BTS, and an isolation mechanism based on traffic shaping decoupled from the BTS. This idea makes the solution feasible and independent of hardware. However, network components need to be modified to achieve control, data tunneling, and isolation.

In [17], the authors considers the optimization of data block allocation as a knapsack problem. The QoS requirements of traffic are included through the utility theory, where the utility function provides a measure for the traffic that needs to be scheduled and the data blocks to be allocated. For the obtained two-dimensional geometric knapsack problem, a heuristic solution is proposed to evaluate different design options, and to evaluate the performance based on the data rate and queuing delay.

Most existing wireless resource allocation methods still have some shortcomings. Firstly, the method based on the minimum and maximum resource reservation in [18] is only aimed at the needs of slices, which may cause the user's quality of service to be unsatisfactory in the case of limited resources. Secondly, many existing algorithms like [19] or NVS in [15] do not take the dynamics of the network into account, the resource allocation is static, and the resource allocation ratio cannot be dynamically adjusted according to actual network requirements. Finally, most algorithms only consider the scene of a single homogeneous slice, and do not consider the scene of multiple heterogeneous slices.

2.2 Deep Reinforcement Learning Theories

In this era of ultra-high-speed development of artificial intelligence (AI), some of the AI technologies could also be used for resource allocation problem in wireless networks to improve

resource utilization and user QoS satisfaction.

2.2.1 Deep Learning

Deep Learning (DL) is a branch of machine learning. It uses multiple processing layers composed of multiple non-linear changes to abstract data at a high level [20]. Deep learning can be regarded as a kind of neural network, which is inspired by brain neurons, especially the framework based on artificial neural network (ANN).

In a neural network, each layer includes multiple neurons. The neurons in the neural networks gives the output by calculating the inputs with certain functions. A neural network can contain multiple hidden layers, and the output of each layer can be used as the input of the next layer.

2.2.2 Reinforcement Learning

Reinforcement learning (RL) [21] is also one of the branches of machine learning, which is trained while continuously interact with the environment. In many machine learning algorithms, learners must try to explore which actions can produce the greatest return instead of being told which action to take. Fig.3 shows the relationship of the branches of machine learning: where in RL, the learner called Agent will be put in a certain environment.

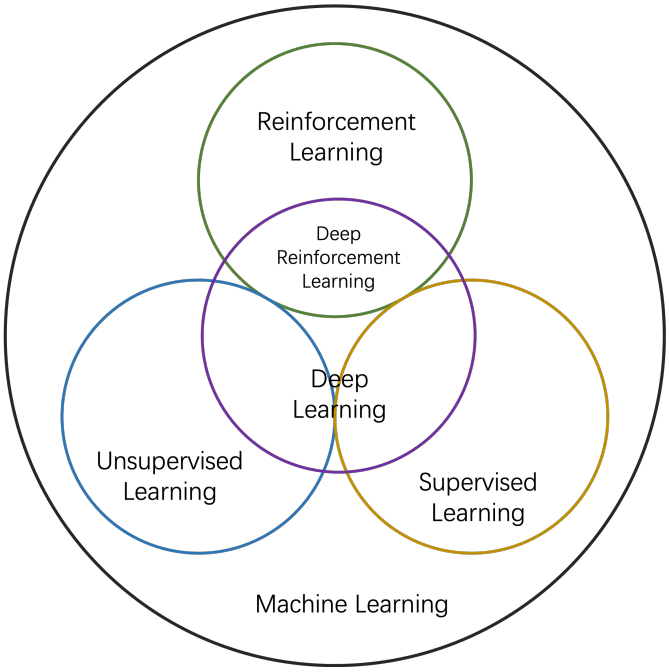


Fig.3: Branches of Machine Learning

At any time, the environment is always in a certain state. The Agent can select a group of actions from the action set, and the state will change accordingly and feedback a reward of this action.

If the Agent wants to obtain the highest reward, it must select the action with the highest

return from the past actions. But in order to discover these actions, the agent needs to explore. Reinforcement learning also has another characteristic: it considers the interaction between the Agent and the external environment. Therefore, it cannot be split into many sub-problems for solving. It can be solved in the reverse direction, that is, by adopting an interactive target search method.

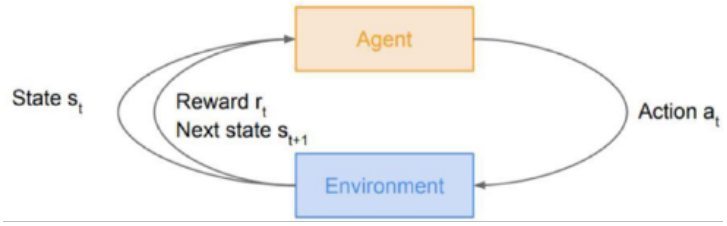


Fig.4: Reinforcement Learning Processing

Reinforcement learning is described based on Markov Decision Process (MDP). As shown in Fig.4, at time t, the Agent is in the state s_t and performs a certain action a_t. And the environment will generate a reward r_{s_t,a_t} and enter the next state s_{t+1} according to the transition probability $P_{s_t,s_{t+1}}(a)$. This reward not only considers the immediate rewards currently received, but also considers the potential rewards in the future. However, the future rewards are less reliable, so the potential rewards need to be multiplied by a discount factor $0 \leq \gamma \leq 1$. The goal of the Agent is to find an optimal action $a^* = \pi^*(s) \in A$ at the current state according to the sum of immediate reward and cumulative discounted reward. According to the definition, the cumulative discounted reward at state s is:

$$V_s^{\pi} = E\left[\sum_{t=0}^{\infty} \gamma_t \cdot r_{s_t,a_t} | s_0 = s\right] \tag{1}$$

where E is the expectation. According to the nature of MDP, the state at the next moment is only determined by the current state and does not relate to the previous states. Therefore, the value function can be expressed as:

$$V_s^{\pi} = R_{s,\pi(s)} + \gamma \cdot \sum_{s' \in S} P_{ss'}(\pi(s)) \cdot V_{s'}^{\pi} \tag{2}$$

where $R_{s,\pi(s)}$ is the average of immediate reward $r_{s,\pi(s)}$, $P_{ss'}(\pi(s))$ is the probability that the system state transfers to s' after executing strategy $\pi(s)$ at state s. Then (2) could be rewritten according to Bellman equation:

$$V_S^{\pi^*} = \max[R_{s,a}] + \gamma \cdot \sum_{s' \in S} P_{ss'}(a) \cdot V_{s'}^{\pi} \tag{3}$$

When the reward and transition probability are unknown as in the case of this thesis, Q-Learning is one of the representative algorithms in RL. Q function $Q_{s,a}^{\pi}$ is defined as the total reward the Agent can get after execute action a at state s, that is:

$$Q_{s,a}^{\pi} = R_{s,a} + \gamma \cdot \sum_{s' \in S} P_{ss'}(a) \cdot V_{s'}^{\pi} \tag{4}$$

and the maximum is:

$$Q_{s,a}^{\pi\,*} = R_{s,a} + \gamma \cdot \sum_{s' \in S} P_{ss'}(a) \cdot V_{s'}^{\pi\,*} \tag{5}$$

then the target of algorithm is to find an optimal strategy to get maximum Q value, and Q value is often calculated from system information $\{s, a, r, s'\}$:

$$Q_{s,a}(t+1) = Q_{s,a}(t) + \alpha\left(r + \gamma \max a' Q_{s',a'}(t+1) - Q_{s,a}(t)\right) \tag{6}$$

where $\alpha \in (0, 1)$ is learning rate, which is the connected to the speed of convergence of $Q_{s,a}$.

The basic idea of Reinforcement Learning that the Agent choose actions according to environment state is similar to dynamic network resource management. However, the network states are varying, and the state space is too large for the Q-value to converge. Therefore, deep reinforcement learning is used to solve the problem in this thesis.

2.2.3 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is an emerging artificial intelligence algorithm framework proposed by Google's DeepMind team [22], whose main role is to learn decision-making. DRL combines the advantages of both deep learning and reinforcement learning.

For DRL problems, it is not needed to prepare any training data before the training starts. The data is obtained through gradual sampling, and this sampling data does not carry any labels.

The deep Q-learning network algorithm is proposed by Google' s DeepMind team for solving control strategy problems. They used this algorithm framework in the Atari 2600 game [23] and trained a network by inputting a large number of screenshots of the game screen, and the output of the network is a value function, which is used to make decisions. DQN uses a neural network to replace the Q table in the Q-Learning algorithm, so that it can perform better in large-scale continuous state scenarios. RL is different from traditional supervised learning. It is a label-delayed learning problem. In order to convert the RL problem into a training problem using supervised learning, DeepMind team proposed a memory replay mechanism.

DQN uses neural network to estimate Q-value, that is:

$$Q(s, a : \theta) \approx Q^{*}(s, a) \tag{7}$$

where θ is the parameter of the neural network. While the estimated Q-value has difference with the Q-value calculated by Bellman equation, a loss function is introduced to minimize the approximation error:

$$L(\theta) = E\left(r + \gamma max\ a' Q(s', a') - Q(s, a : \theta)\right)^2 \tag{8}$$

2.3 Software Defined Network

The realization of network slicing relies on Network Function Virtualization (NFV) and Software Defined Network (SDN) [24]. NFV technology softwareizes each protocol layer function in the communication network, making it a Virtual Network Function (VNF), so that no proprietary equipment is needed to realize this function [25]. SDN is an emerging network architecture which realizes the separation of control and data, so that the resources of the entire network can be easily managed, and resources can be flexibly allocated to each VNF to meet different requirements [26].

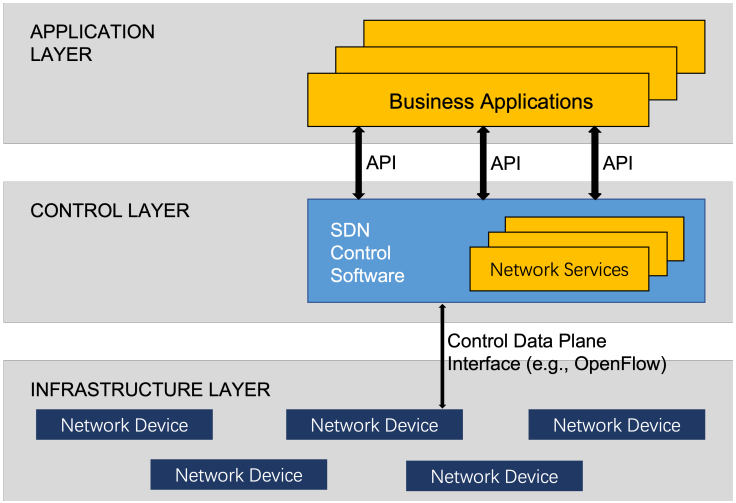


Fig.5: SDN Architecture [27]

As shown in Fig.5, defined by open network foundation (ONF) in 2012, SDN mainly includes three layers: the upper application layer, the control layer, and the infrastructure layer. The function of the upper application layer is to present network services and abstract network models; the function of the control layer is similar to the network operating system, mainly to manage network resources; the function of the infrastructure layer is packet switching.

Besides the separation of control and data, SDN can also deploy services more quickly, and the configuration process of services is simplified, network capacity can be flexibly configured, and network operation efficiency has been greatly improved. The wireless virtual network resource management system proposed in this thesis is implemented based on SDN architecture, and the SDN controller is equivalent to the Agent in the DRL.

3 System Model

3.1 System Description

In the future 5G network, due to the diversification of services, multiple heterogeneous slices will coexist and share the wireless resources on the same base station. This thesis builds an automatic wireless virtual network resource management system based on DRL to maximize the resource utilization of the entire system while ensuring user QoS satisfaction.

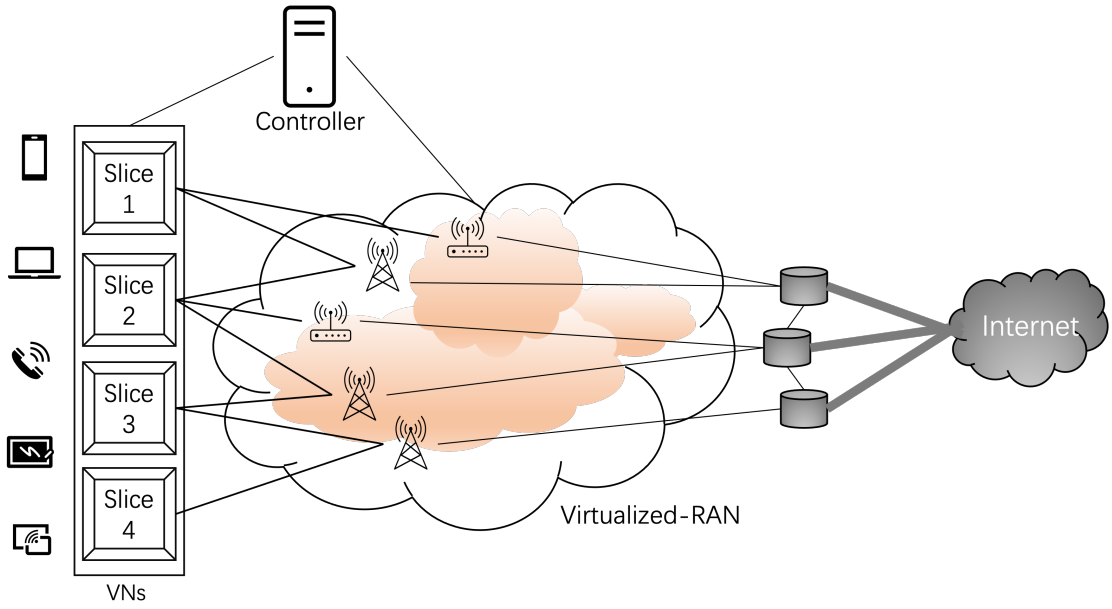


Fig.6: System Description

As shown in Fig.6, the system adopts the SDN network architecture, where the controller is used to make decisions and control the base station (BS) to adjust the resource ratio between slices. The system also includes mobile user equipment (UE), slices, base stations, and resources. Each type of slice provides different services. UEs will be connect to a certain BS and then request the resources of the slice provided by BS. The network resources in this thesis are the radio resources on the radio access network (RAN) side, which are actually resource blocks (RBs) composed of time domain and frequency domain.

In this thesis, the wireless cell has multiple BSs with all types of heterogeneous slices. Assuming that the operator configures the same proportion of slice resources on each BS, the problem can be simplified to a single BS scenario. For example, there are four types of slices in total: slice1, slice2, slice3 and slice4, the initial resource ratio on the BS is 1:1:1:1, that is, each slice occupies 25% of the resources on the base station. After a period of time, the resource utilization of each slice on the BS can be measured, slice1: 60%, slice2: 100%, slice3: 30%, and slice4: 80%. If the number of UEs in slice 2 increases at this time, its resources will

be insufficient, thereby reducing the QoS of users. However, the resource utilization of slice1 and slice3 is relatively small. If the number of their UEs does not change, it will cause a waste of resources, hence the decrease of resource utilization. Therefore, the goal of this system is to ensure the QoS of users and maximize the resource utilization of the entire network in the case of limited resources.

3.1.1 Slice-level Resource Allocation Problem

When multiple heterogeneous slices coexist, the slice resource allocation ratio on the BS is a very important parameter, which affects the performance of the entire network. If the traffic of the UE of the network is always fixed, only one initial allocation is required. However, the real scenario is that the number of UEs requesting services for various slices will change, so the slice resource utilization on the BS will change accordingly. Therefore, it may happen that some slices have a lot of resources left, and some slices have far from enough resources.

Therefore, the problem goes to be that how to meet the resource requirements of slices by adjusting the resource ratio between slices on the BS when the total resources are sufficient. Reinforcement learning such as Q-learning is a appropriate solution to solve such a controlling problem. However, the Q-Learning algorithm is only suitable for discrete state space scenarios, and cannot be well solved for continuous state space scenarios. Therefore, automatic adjustment of slice resources can be achieved through deep reinforcement learning algorithms, such as DQN algorithm.

As shown in Fig.7, after the Agent allocating the resources to slices, the system should map the resource ratio to BSs.

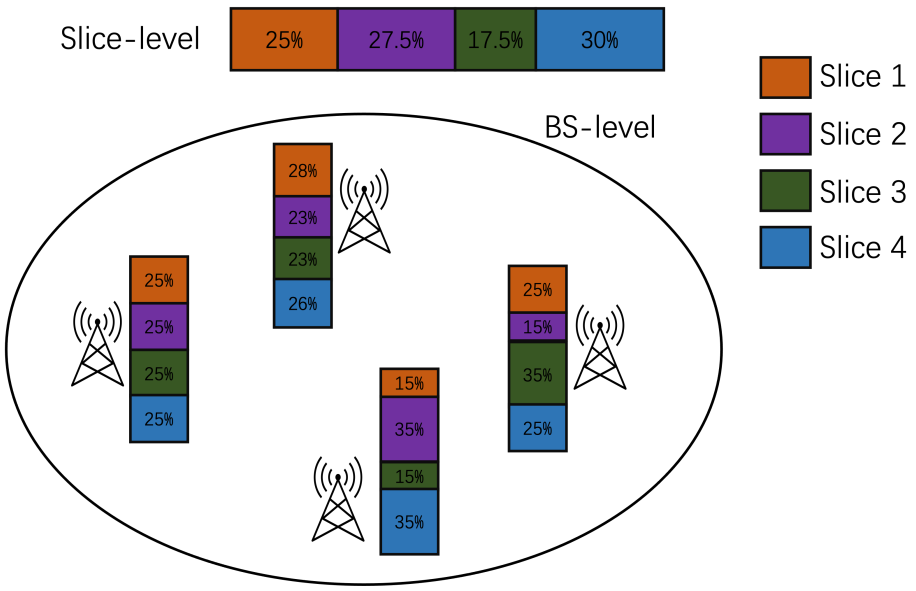


Fig.7: Mapping of Resource Ratio

3.1.2 User-level Resource Allocation Problem

The physical resources on the RAN side are limited, in order to maximize the utilization of resources, it is also an important issue to map the resource from slices to BSs and then to users. Traditional method is to divide the bandwidth equally, that is, to divide the entire system bandwidth by the total number of data streams. However, the transitional way does not consider the users' QoS requirements, which may lead to dissatisfaction with the QoS requirements.

In LTE, the channel bandwidth can be divided into physical resource blocks (PRBs) [28], and each PRB occupies 180 kHz in the frequency domain and 0.5 ms in the time domain. One frame of LTE is 10ms, and each frame has 10 subframes, one of which is 1ms, which is the scheduling time unit of LTE, called transmission time-interval (TTI). In 5G, one PRB contains resources similar to LTE. Therefore in this thesis, the settings about PRBs could also be considered as in LTE. Therefore, the entire channel bandwidth is like a resource grid composed of PRBs, and the resources actually allocated to the UE are these PRBs.

3.2 Problem formulation

In this thesis, a set of BSs is considered such that $k \in \{1, 2, \ldots, K\}$ with different transmitting power denoted by P_n and each BS a set of slices $s \in \{1, 2, \ldots, S\}$. Slices have QoS requirements $d \in \{d_1, d_2, \ldots, d_S\}$. For each slice s, there is a set of users $U_s \in \{U_1, U_2, \ldots, U_S\}$. Assuming the QoS requirements of users on each slice are the same, the total system bandwidth is W, bandwidth allocated to each slice is $w \in \{w_1, w_2, \ldots, w_S\}$.

The definition of objective function of the resource allocation among slices as:

$$arg_w \max E\{R(w,d)\} = arg_w \max E\{\mu \cdot Sat(w,d) + \xi \cdot RU(w,d)\} \tag{9}$$

$$s.t. \quad w_1 + w_2 + \cdots + w_S = W, \quad w \in \{w_1, w_2, \ldots, w_S\} \tag{9a}$$

$$d \in \{d_1, d_2, \ldots, d_S\} \tag{9b}$$

$$d_i \quad \text{Certain Traffic Model}, \quad \forall i \in [1, \ldots, S]$$

where $Sat(w,d)$ is the system satisfaction, $RU(w,d)$ is the resource utility rate, $0 \leq \mu \leq 1$ and $0 \leq \xi \leq 1$ are the respective importance. Constraint (9a) means system allocates all the bandwidth resource to the slices, (9b) indicates the QoS requirements, which can be either data rate or delay requirements.

The resource ratio each slice occupied could be calculated by using:

$$V_s = \frac{w_s}{W} = \frac{w_s}{w_1 + w_2 + \ldots + w_S} \tag{10}$$

The transmit rate of user u connected to BS k on slice s can be expressed as:

$$c_{sk}^u = \frac{W}{K} \cdot log_2\left(1 + \frac{S}{N}\right) \tag{11}$$

And the transmission delay of this user is expressed as:

$$\tau_{sk}^{u} = \frac{1}{c_{sk}^{u} - \lambda_u} \tag{12}$$

where λ_u is the data packet arrival rate in M/M/1 queuing model which the data flows obey.

In order to map the system-level resource to slices, the transmit rate is used to calculate the weight of each BS to corresponding to the system, then the ratio of resource that slice s occupied on BS k is:

$$V_{sk} = K \cdot V_s \cdot \frac{\sum_{u \in u_{sk}} c_{sk}^{u}}{\sum_k \sum_{u \in u_{sk}} c_{sk}^{u}} \tag{13}$$

While (12) might cause the total resource ratio on some BS is not 1, V_{sk} should be normalized it by using:

$$V'_{sk} = V_{sk} \cdot \frac{1}{\sum_s V_{sk}} \tag{14}$$

Hence, the actual bandwidth that slice s occupied on BS k is:

$$w_{sk} = \frac{W}{K} \cdot V'_{sk} \tag{15}$$

After the resources are allocated to slices on BSs, BS will allocate them to data flows sent by users. Many systems define the resource granularity of virtualization as time slot and bandwidth ratio on the RAN side [29], but this thesis considers RB-level resource virtualization. The bandwidth of each BS is B, it can be divided into M RBs in the frequency domain, and the bandwidth of each RB is B_m. A scheduling frame is divided into T subframes, the length of each subframe is t_l , then the length of a scheduling frame is $T \times t_l$. According to Shannon's formula, the transmission rate that can be obtained by assigning an RB to the UE is shown as:

$$c_{uk}^{(t,m)} = \frac{B_m}{T} \cdot \log_2^{1+SINR_{uk}} \tag{16}$$

where $SINR_{uk}$ is the signal-to-noise-ratio between UE u and BS k. In order to improve the QoS satisfaction of users, it is necessary to carry out effective physical resource allocation. Also, in order to ensure that the actual delay of each user u served by the slice is less than the specified maximum delay requirement, the interval between two consecutive data packets sent by the user must be less than $Td_u^{\max}$. Therefore, the number of time slots and frequency domain RBs required for the flow sending can be calculated by the user according to the user's QoS requirements (minimum transmission rate and maximum transmission delay):

$$n_u^h = \left\lceil \frac{T \times t_l}{Td_u^{\max}} \right\rceil \tag{17}$$

$$n_u^v = \left\lceil \frac{c_u^{\min}}{n_u^h \times c_{uk}^{(t,m)}} \right\rceil \tag{18}$$

where n_u^h is the minimum required time slots and n_u^v is the minimum required RB on frequency domain. Hence, the transmission rate that can be obtained by every user u from BS k is:

$$c_{uk} = n_u^h \times n_u^v \times c_{uk}^{(t,m)} \tag{19}$$

From the equations above, it can be calculated that the actual number of PRBs required by the flow of each UE is $n_u^h \times n_u^v$. Therefore, the physical resource allocation problem can be modeled as a two-dimensional knapsack problem and can be solved by using the method in [30]. Which is not the main topic in this thesis. Through the PRB resource allocation, it can maximize the utilization of resources in the case of limited resources, thereby reducing the idleness of resources.

4 Resource Allocation Algorithm Based on DQN

In DeepMind's paper [23], the states of the game environment can possibly be up to 1067970. If the Q-Learning algorithm is used, then the Q table needs to have 1067970 rows, which is impossible to achieve. Even with such a large memory, it will take a considerable amount of time to look up the table, so the Q-Learning algorithm cannot adapt to scenarios with large state space.

In order to solve the problem of such a large state space, deep neural networks can perform feature extraction on highly structured data, thereby replacing the Q-table in the Q-Learning algorithm. Therefore, this thesis uses a deep neural network to fit the Q-function, and the input state and action can output the corresponding Q-value. We can also just input the state, and the output is the Q-value of all possible actions. The advantage of this approach is that it only takes one forward pass through the network to output the Q-values for all possible actions.

DQN algorithm is used to adjust the slice level resource dynamically. The purpose of this strategy is to balance the resource allocation ratio among the slices as much as possible to maximize the resource utilization of the entire system under the premise of ensuring user QoS. Therefore, this thesis first builds a model for resource allocation between slices. Since the resource allocation between slices is for the resources of the whole system, it is necessary to map the resource ratio of the slice level to each BS, so a detailed resource ratio mapping model is given and it can be solved by the algorithm in [30].

The three basic elements of DQN (state, action, and reward) are defined as follows:

State: as to maximize the resource utilization of the entire system while ensuring user QoS satisfaction, state mainly includes three quantities: the resource reservation ratio of the slice V_s, the resource usage ratio of the slice RU_s, and the average user QoS satisfaction of the slice Sat_s. RU_s refers to the ratio between the resources actually used and the reserved resources. Sat_s is the result of the feedback after physical resource allocation calculated as:

$$Sat_s = \frac{1}{U_s} \cdot \sum_{u \in U_s} Sat_u \tag{20}$$

where Sat_u is the satisfaction of user u, which is:

$$Sat_u = \frac{1}{1 + e^{-\left(c_{sk}^u - c_u^{min}\right)}} or \frac{1}{1 + e^{-\left(\tau_{sk}^u - \tau_u^{max}\right)}} \tag{21}$$

c_u^{min} and τ_u^{max} are the data rate and delay constraints for two types of slices, c_{sk}^u and τ_{sk}^u are the actual data rate and transmission delay.

Hence the system state is defined as $s = [V_s, RU_s, Sat_s]$.

Action: In DQN, in order to avoid getting stuck in a local optimum, the ϵ -$greedy$ algorithm is used to choose actions. ϵ -$greedy$ algorithm choose a random action with a probability ϵ,

and choose the action with maximum Q-value with probability $1 - \epsilon$. Actions are defined as $a = \{-0.4, -0.2, 0, 0.2, 0.4\}$, which contains the adjustments of resource reservation. Too large action space will slow down the convergence speed, that is why this thesis uses a small action set.

Reward: In each episode, environment will return a reward to measure the action taken by Agent, positive reward indicates that action is optimal and negative shows it is not, that is:

$$reward_s = w_1 \cdot (Sat_s - 0.5) + w_2 \cdot (RU_s - 0.5) \tag{22}$$

where $0 \leq Sat_s \leq 1$ and $0 \leq RU_s \leq 1$ represents the weight of satisfaction and resource utility, respectively.

DQN process: In [23], DeepMind use the loss function below to update Q-value:

$$L = \frac{1}{2} \cdot [r + \max_{a'} Q(s', a') - Q(s, a)]^2 \tag{23}$$

where $r + \max_{a'} Q(s', a')$ is the target Q-value, $Q(s, a)$ is the prediction Q-value. Therefore, the updating of Q-table in Q-Learning is equal to the update of network parameters in DQN.

DeepMind also introduces memory replay as an important technique for the DQN training. All experiences generated during the game are stored in the memory pool, and the capacity of the memory pool is limited. If the experience is full, old experience will be replaced with new experience, each of which is a transition $\{s, a, r, s'\}$. In order to avoid correlation between samples, the training network is trained by randomly taking a mini-batch of data from the memory pool. If the most recent experience is used every time, the network may get stuck in a local optimum.

For each transition, the system takes following steps to train the network:

(1) Input the current state s, output the prediction Q-Value of all possible actions;
(2) Enter the next state s', then output the Q-value of all actions in this state, and find the largest Q-value $\max_{a'} Q(s', a')$;
(3) Calculate the target Q-Value by $r + \max_{a'} Q(s', a')$, then set the Q-value of action a as the target Q-Value, and set the Q-value of other actions as prediction Q-Value;
(4) Finally, update the weights of the network through back propagation.

While carrying out the action selection through the ϵ -*greedy* algorithm, the action process of selecting the maximum Q-value is called exploiting, and the process of randomly selecting actions is called exploration. The choice of this probability ϵ could affects the convergence time. During training, the probability of exploiting is generally much larger than the probability of exploration. For example, when ϵ equals to 0.05, that is, to explore with a probability of 5%, and exploit with a probability of 95%. However, in the process of algorithm implementation, the purpose is to make the algorithm to converge, so the DQN can continuously reduce the probability of exploration.

In the ideal situation, after training, the system loss will eventually converge, that is, the parameters in the neural network will not change. Therefore, after the current state is directly input, DQN will output the Q-value of all possible actions and find the action with the largest Q-value, which is the optimal action corresponding to the current state.

The detailed flow of the proposed method is shown in Fig.8: where slice/user admission

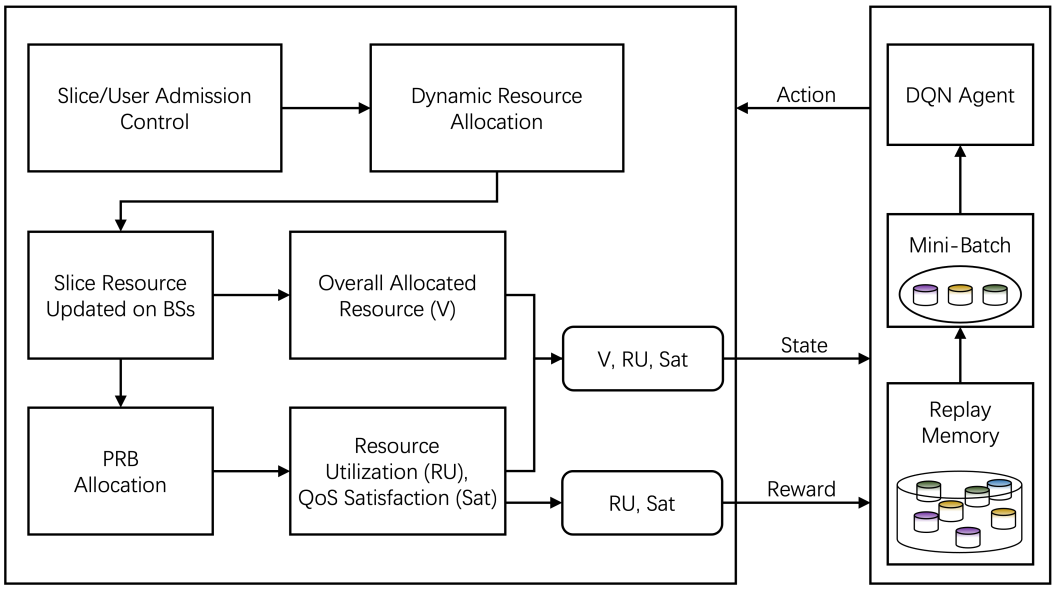


Fig.8: DQN Based Resource Allocation Flow

control is to maximize the number of UE access by connecting the users to BSs. The strategy applied is to connect the users to the closest BS. The strategy might cause the overload of BSs, but since it is assumed that the users are distributed uniformly in the environment, this problem could only happen in a small probability.

After the users are connected to corresponding BS, the initial resource allocation directly allocate a fixed resource ratio to the slice, and then map the entire resource ratio to each BS. If this ratio remains fixed, once the number of UEs in the network scenario changes, it may lead to insufficient resources or excess resources. In order to adapt to this dynamic network scenario, it is necessary to dynamically adjust the resources of the slices according to the state of the current environment.

Since the resource ratio (V) of the slice is mapped on the BS, physical RB allocation needs to be performed for the data flow of the UE. The RBs on each BS are limited, and the scheduling method also affects the resource utilization (RU) and user QoS satisfaction (Sat). On the premise of ensuring user QoS satisfaction, the utilization rate of resources should be maximized, so that the probability of resource idleness could be minimized.

Finally, each time the Agent performs resource allocations between slices and allocates physical RBs to the UE, the Agent will receive a reward consisting of user QoS satisfaction and resource utilization, and the environment will also enter the next state. Due to the memory

replay mechanism of DQN, the quadruple consisting of state, action, reward and next state is stored in the replay memory pool. If the pool is full, Agent will randomly select a batch of data from the memory pool for training, so as to continuously update the parameters of the neural network to reduce the system loss.

Combing with the wireless network scenario in this thesis, the DQN-based resource reservation algorithm is detailed in Algorithm 1.

Algorithm 1: DQN-Based Resource Allocation Algorithm

```
Set replay memory pool size D, mini-batch size d, the discount factor γ and epsilon ϵ
Set the dim of states and actions
Initialize Q-network with random weights
while true do
    Collect state s ← [V, RU, Sat]
    Generate a random number π
    if π< ϵ then
        Random generating action a
    else
        Input the state to the network and select the action that has maximum Q-value
    end
    Update the slice-level resource and calculate reward r
    Update the resource fraction V_sk in all BS
    Convert the BS-level resource fraction to the slice-level resource fraction
    Store transition {s, a, r, s'} in the replay memory pool
    if replay memory is full then
        Pick a mini-batch of samples from the replay memory for all samples do
            Calculate prediction Q-values by DQN
            Update target Q-value by r + maxa' Q(s', a')
            Train DQN by the loss function L = 1/2 · [r + max_a' Q(s', a') − Q(s, a)]^2
        end
    end
end
```

5 Simulation & Result Analysis

In order to evaluate the performance of the DQN-based resource allocation method between slices proposed in this thesis, the researcher configures the network scenarios of the simulation experiments. The network scenarios in this thesis can be divided into two categories: one is a single-slice network scenario, and the other is a multi-slice network scenario.

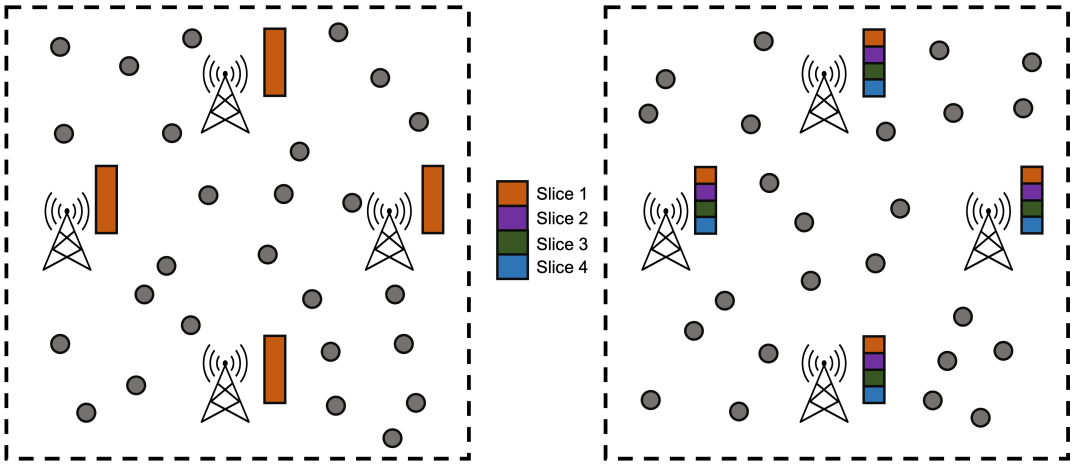


Fig.9: Simulated Network Scenarios

As shown in Fig.9, the left side is a single-slice network scenario, in which 4 BSs are deployed, the same type of slice is deployed on each BS and only one slice is deployed on each BS. UEs are uniformly distributed around the four BSs. For single-slice network scenarios, simulation experiments are used to verify whether DQN will adjust the allocated resources of slices according to the state of the network. The main evaluation parameters are the change of state, action, and reward in single episode, the change of loss, both allocated and used resource ratio change, and the change of system reward.

The right side is a multi-slice network scenario, in which four BSs are also deployed, and four types of slices are deployed on each BS. The allocated resource ratio of slices on each BS may be the same or different. It is related to the number of UEs associated with the BS, and the UEs are uniformly distributed around the four BSs. For multi-slice network scenarios, the performance of the proposed algorithm will be evaluated by comparing the other two resource allocating algorithms. The main evaluation indicators are: system resource utilization, system satisfaction, the change of resource allocation ratio and usage ratio.

The other simulation parameters are shown in Table 1.

5.1 Single Slice Situation

In the single slice scenario, each of 4 BSs has one slice and the initial allocated resource ratio is 0.1. The replacement period of target network parameters is 100 steps. In each episode, the

Table 1: Simulation Parameters

Parameter name	Value	Parameter name	Parameter
Slice types	1 or 4	Packet arrival rate	100 (packet/s)
BS numbers	4	Packet size	[eMBB:400, HdBB:4000, uRLLC:120, mMTC:500] (bit)
Number of users	60 - 430	Packet arriving time	uRLLC: uniform, others: exponential distribution
System bandwidth	20 MHz	Min rate constraint	[eMBB:150, HdBB:800, uRLLC:10, mMTC:60] (kbps)
BS transmit power	30 dBm	Max delay constraint	[eMBB:100, HdBB:120, uRLLC:10, mMTC:105] (ms)
BS covering radius	150 m	Slicing period	200 ms
Noise density	174 dBm/Hz	Replay memory size	8000
Learning rate α	0.01	Mini-batch size	32
Discount factor γ	0.9	$\epsilon - greedy$ possibility	0.04

number of users will change 10 times, that is, Agent will do 10 actions according to the state. Since the goal of this simulation is to prove that DQN can appropriately adjust the allocated resource ratio, the comparison will not be taken.

As shown in Fig.10, it is the detail figure in the final episode of DQN, which includes Allocated (allocated resource ratio), Used (resource ratio used by users), Action (action taken by Agent), Sat (satisfaction of user QoS), RU (resource utility), and reward. From the figure, it can be seen that as the number of user increases, DQN will adjust the resource ratio, and the allocated and used resource ratio almost coincide, which means the resource utility is in a high level. The user QoS satisfaction is always above 0.5, which means the adjustment of

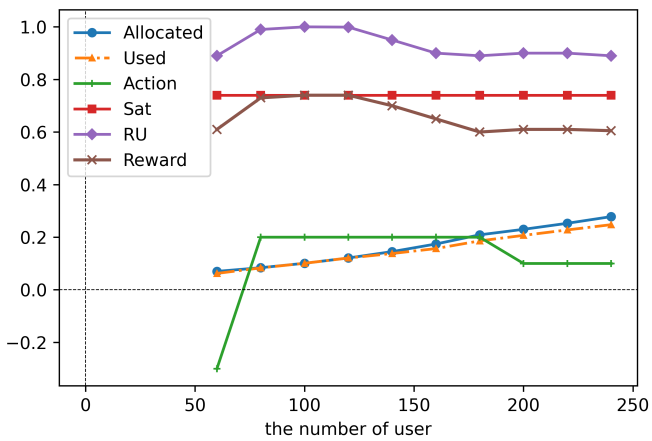


Fig.10: Details of a Single Episode

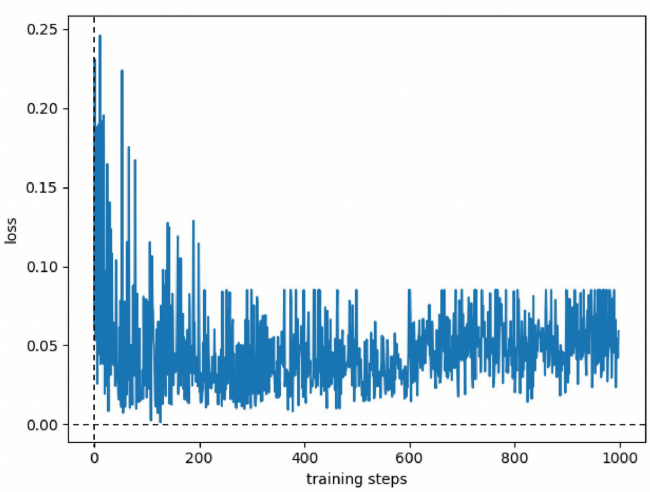
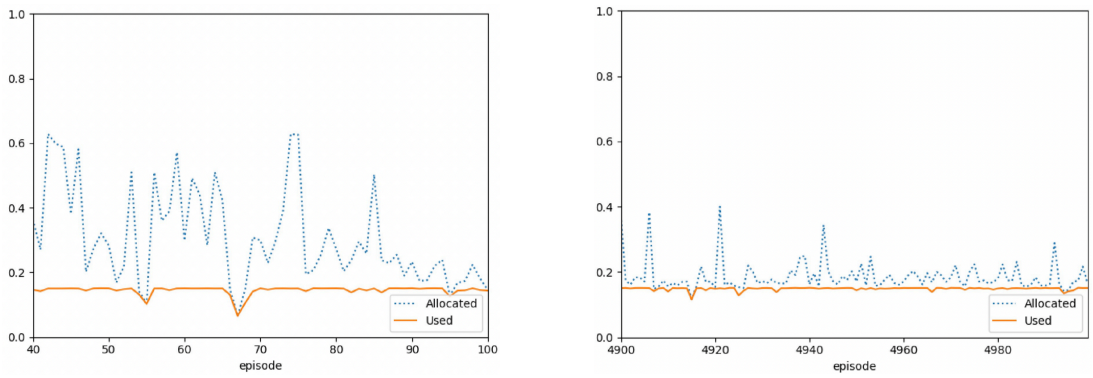


Fig.11: Loss Change in Single-Slice Scenario

resource allocation by DQN is appropriate. Therefore, the strategy based on DQN can adjust the resource allocation ratio according to the system state to increase resource utility under the premise of user QoS satisfaction ensuring.

Fig.11 shows that the DQN loss decreases as the training step increases and converges within a value, which means that the Agent will execute the optimal action eventually.



(a) (b)

Fig.12: Change of Allocated and Used Resource Ratio

At the early period of training, as shown in Fig.12 (a), allocated resource and used resource had big difference, which means the resource utility was low. While at the late period, as shown in Fig.12 (b), the difference reduced, that is, the utility increased. Therefore, as DQN training, the allocated resource ratio will gradually fit the needed resource ratio and increase the resource utility.

In summary, for the single-slice network scenario, through the analysis of the simulation results, the DQN-based resource allocation scheme proposed in this thesis will dynamically adjust the resource allocation ratio of the slice according to the current network load, and can Improve the resource utilization of the system, thereby reducing the waste of resources.

5.2 Multi-Slice Situation

For multi-slice network scenarios, the algorithm will be applied in the environment with four types of network slices in the same system, namely eMBB, HdBB, uRLLC, and mMTC. HdBB indicates that enhanced communication for high-definition images or videos such as high-definition digital television live stream or VR/AR information, which requires faster date rate than traditional eMBB services.

In order to verify the effect of the algorithm proposed in this thesis, the performance is compared with two other slice resource allocation methods. One is called hard slicing. Hard slicing means that each service is always allocated with $\frac{1}{4}$ of the whole bandwidth (because there are four types of services in total) and round-robin scheduling is conducted within each slice. The other one is the algorithm proposed in [15], called NVS. NVS refers to dividing the resource ratio of slices according to the rate requirements of slices. In the experiment on Section 5.1, since the action of DQN is to adjust the ratio of resources at the slice level, the sum of the total resources of the four slices after adjustment may be less than 1. Both hard slicing and NVS use all system resources. For the fairness of the comparison, a normalization operation is required to make the sum of the total resource ratio of all slices equal to 1.

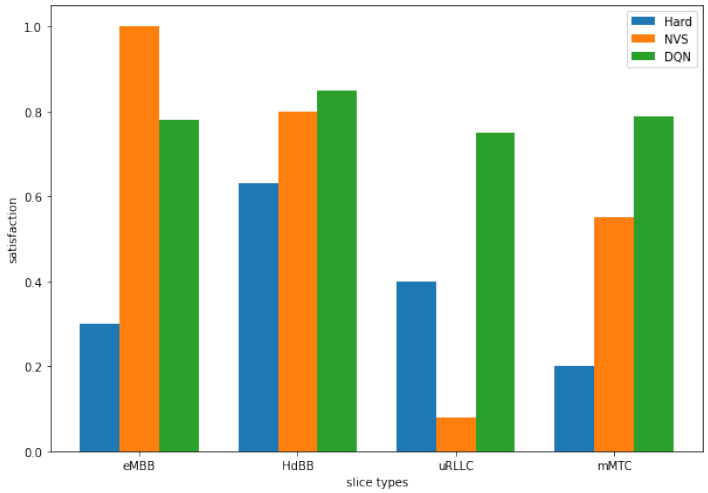


Fig.13: Satisfaction Comparison of Slices

As shown in Fig.13, it is a comparison of slice satisfaction of the three methods. As can be seen from the figure, while using hard share, the satisfaction are always below 0.5, which means that the allocation strategy to give all the slices the same ratio of resources is not appropriate to the new form of network service. The satisfaction of eMBB, HdBB and mMTC slices of NVS is above 0.5, but the satisfaction degree of uRLLC slice is less than 0.1, which shows that the resource allocation of uRLLC slice is unreasonable. The satisfaction of the four slices while applying DQN can keep above 0.5. And the satisfaction of the four type of slices doesn't vary too much. In summary, the DQN-based resource allocation algorithm proposed

in this thesis has a good balance, and it can ensure the satisfaction of multiple slices. NVS can only guarantee the satisfaction of a part of the slices, and the balance is not as good as the DQN method.

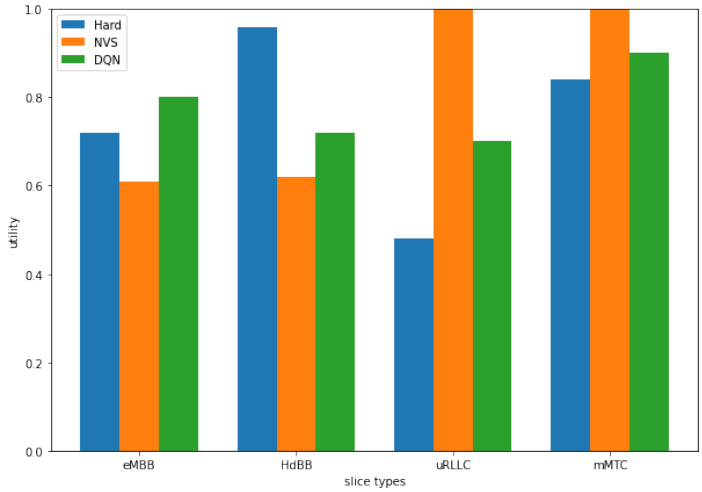


Fig.14: Resource Utilization Comparison of Slices

Combining Fig.13 with Fig.14, it can be seen that even the hard share method has high resource utilization, the utilization of uRLLC is below 0.5, which means it cannot ensure the balance of the slices. The satisfaction of the eMBB slices of DQN and NVS methods is above 0.5, and the satisfaction of the HdBB slices of the three methods is the closest, but the overall resource utilization of the DQN-based method is higher than the other two methods. NVS's resource utilization for uRLLC and mMTC slices is 1, which indicates that its resource allocation is inappropriate.

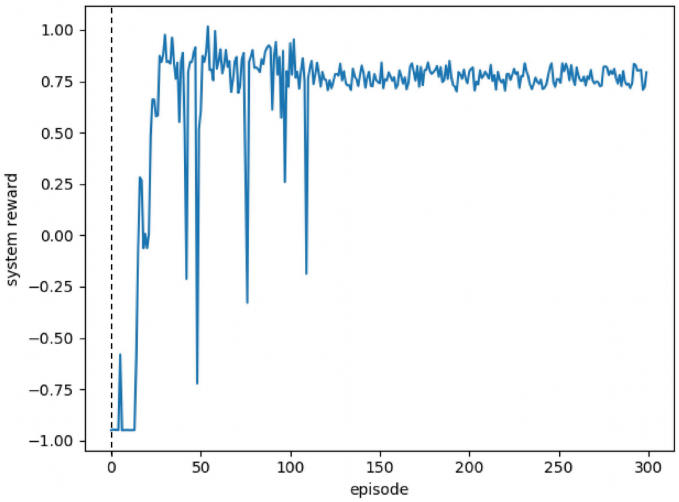


Fig.15: Change of System Reward

For multi-slice network scenarios, the change of system reward is shown in Fig.15. In the first 50 episodes, the system reward showed an upward trend, which shows that DQN is constantly

optimizing the network parameters. Between the 50th and 110th episodes, the reward will still have a large oscillation, which indicates that the network parameters of DQN have not reached the optimum, and they are still learning from the experiences. From the 110th to the last episode, the system reward has been fluctuating around 0.75, and the fluctuation range is very small, which shows that the network parameters of DQN have basically stabilized, and there will be no major changes.

5.3 Discussion

In this section, simulation experiments are carried out for single-slice network scenario and multi-slice network scenario. In single-slice network scenario, through analysis of the simulation results, the scheme proposed in this thesis can dynamically adjust the resource allocation ratio of the slice according to the current network load situation, and can improve the resource utilization of the system, thereby reducing the waste of resources. While when the algorithm is applied in the multi-slice network scenario, compared with the hard slice and NVS method, it has be prove that the proposed method based on DQN can appropriately adjust the network resource allocation to slices while ensuring the system QoS and the resource utilization for the system with four types of slices.

From the simulation results, it could be said that DQN based resource allocation method can be adjusted to the network scenarios that network slicing technology requires. By changing the resource allocation ratio to slices based on the system change, and through the Agent in DQN learning from the interaction with the environment, the DQN can be applied to the network slicing scenarios to efficiently improve the system performance and save the network resources.

To apply DQN in solving the resource allocation problem can be a research topic worth considering. However, this proposed scheme still has some problems that cannot be ignored:

5.3.1 The Acceptation of New Slices

The problem of system changing is not only about the time situation, but also about a more critical situation. In the reality, the network system is always changing, not only in the scheduling queue or user level, but also in the slice level. When a new type of slice is going to join the system, while the other current parameters remains the same, the system should take appropriate actions to make sure that the new slices are allocated enough resource to satisfy the requirements.

The new slice accepting problem will be a major problem to be considered in the future works. The system would be trained to face more complicated situations, which means that system states will be more diverse, and the DQN Agent should be more intelligent to adapt the new situation.

One of the ideas to solve this problem is to also use DQN algorithm based method to adapt

the environment changing and manage the acceptation of new slices. However, if too many isolated DQN systems are nested, the whole system will too complicated to be deployed.

5.3.2 Mobile Scenarios

This thesis only considers the situation that users have fixed location in the system, the UEs just appear or disappear on the environment, which is actually not the same as real situation. It is necessary to consider that users moving in the network, especially when it comes to eMBB and uRLLC users. The eMBB users mostly include mobile devices and uRLLC users might include smart vehicles, whose moving could result in the system complexity increase.

5.3.3 The Time of Calculation

Python was used while training the DQN system in this thesis and run on Google Colaboratory platform with free GPU accelerator. It took around 5 to 15 seconds to calculate one episode, in some extreme situations, system could spend 30 seconds to run the codes, hence would spend several hours to train the whole system. Hence, even DQN have efficiency in resource allocation, the system itself is not efficient enough.

This problem could be severe because that if it is tested in different network scenarios, the Agent should be trained again, which means that every time the system environment changes, it would take a long time to get an optimal result.

This problem can be solved by replacing the operating platform with a high speed computer, or improving the codes to more efficient version.

Besides, the generality of the algorithm should also be evaluated, that is, the algorithm should be adapted to different networks scenarios. With the state, action and/or reward changes, if the proposed method can also get the ideal result to support itself should be fatherly evaluated.

5.3.4 The Evaluation Benchmark

This thesis has compared the proposed algorithm to two other resource allocation methods. Neither hard slice nor NVS uses DQN or RL in their system, which means the evaluation in this thesis is one-sided. The superiority of proposed method should be proved by being compared the performance with other similar methods.

However, the fact is that while several algorithms based on DQN or RL was proposed, their optimizing goals are almost different, it is hard to compare their performance. It is also a subject that can be discussed in the future to unify the evaluation standards of resource allocation or management problems.

5.3.5 The Stability and Efficiency of DQN

As being mentioned in section 4, in some situations, DQN will fall into local optimum, that is, the system can get a solution that is optimal within a neighboring set of candidate

solutions, but is not optimal among all possible solutions. Which will result in misjudgment to the system performance.

If the chosen hyper parameters are not appropriate, DQN may become not stable enough to get the optimal solutions in all situations. If the local optima appears, the system would not converge to the target value.

This problem could be solved by applying some of the advanced DQN algorithms such as Double DQN [31] (DDQN) or Dueling DQN [32].

DDQN get the target Q-values by putting the action with max Q-values from current network into the target network. DDQN can reduce overestimation bias of Q-values calculated by natural DQN used in this thesis, which results in better final policies.

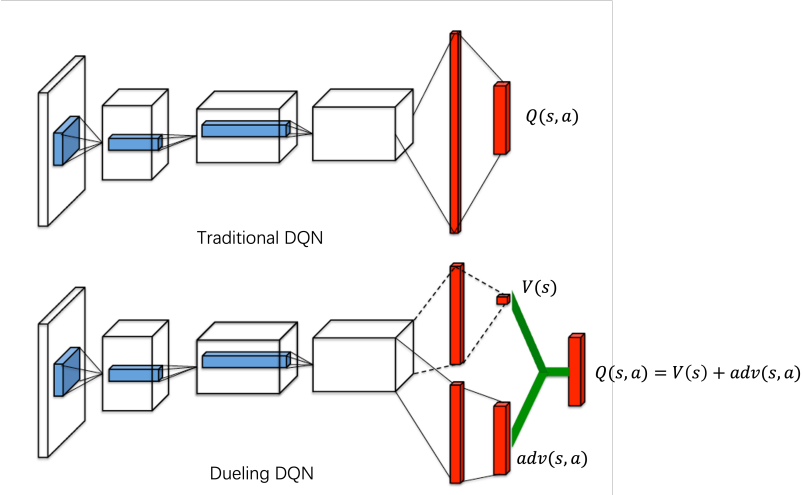


Fig.16: Traditional Q-network (top) and Dueling Q-network (bottom). [32]

While Dueling DQN, which cannot be abbreviated to DDQN, is an advanced as shown in Fig.16, has two streams to separately estimate (scalar) state-value and the advantages for each action. In Dueling DQN, the Q-value is calculated by the combination of the value function $V(s)$ and the advantage function $adv(s,a)$. The value function $V(s)$ tells that how much reward will be gained from state s. And the advantage function $adv(s,a)$ tells us how much better one action is compared to the other actions. Since sometimes it is unnecessary to know the exact value of each action, it is enough for the Agent to just learn the state-value function. Hence, this change can accelerate the training process of DQN.

6 Conclusion & Future Works

6.1 Conclusion

This thesis has proposed a network resource allocation strategy based on DQN, which is aiming at allocating resources for the network slices and assure the user QoS satisfaction. The performance of proposed algorithm was evaluated by simulating in the configured network scenarios.

At the single-slice network scenario, through analysis of the simulation results, the scheme proposed in this thesis is proved to be able to dynamically adjust the resource reservation ratio of the slice according to the current network load situation, and to improve the resource utilization of the system, thereby reducing the waste of resources.

At the multi-slice network scenario, by comparing with hard slice and NVS methods, the DQN-based scheme shows its superiority in dynamic network scenarios. The algorithm can improve the system overall resource utility while ensuring the user QoS.

In the era of 5G networks, network slicing technology will be introduced to virtualize wireless networks. For heterogeneous multi-slice network scenarios, slices will share the resources of the entire physical resource pool. Therefore, resource managing among slices could be a very important issue. Resource ratio divided by traditional methods mostly stays in the same level, which might cause the resources insufficient or being wasted when the number of users changes. While DQN-based method can automatically change the allocated resource according to the network state.

However, the proposed method still has some shortcomings to be improved in the future. After the simulation, it has reviewed these problems and discussed some of the problem-solving ideas.

6.2 Potential Future Works

In the future, as being discussed in section 5.3, the research should focus on the improvement of DQN scheme. With the feature that can adjust solutions according to the change of environment, DQN has the potential to handle the network resource management problems. The potential future works could be summarized as following:

(1) Using DQN-based algorithm to accept users of new type of slices or same kind of slices with different requirements.
(2) The three basic elements of DQN (state, action, and reward) should be defined by the environment changing. It could be considered to dynamically redefine the three elements so that the system efficiency could be further improved.
(3) Considering using GPU to accelerate the calculation of DQN system. This might lead

to increased research costs, it is necessary to balance the gains and sacrifices if it comes to commercial scenarios.

(4) The Agent can be trained by DDQN [31] or Dueling DQN [32] to get more reliable results and improve the system performance.

Acknowledgements

Grateful acknowledgement is made to my supervisor, Professor Hitoshi Aida, for his guidance, enthusiastic encouragement, considerable help, including weekly meetings during the writing of this thesis, sincere gratitude to him for wide range of advice, from the research idea and topic to the directions of research.

I am very grateful to the professors who asked questions and gave my advice in the Rinkos once a semester and the final presentation.

I would also like to thank Ms. Fumiko Kouda for her questions and advice on my research and presentation materials during the meetings. Also, I would like to thank the lab's secretary, Ms. Misako Motooka, who patiently helped me with every cumbersome procedure of visa and other stuff along with being a foreign student, with Mr. Shingo Chiba, the technical specialist of the lab, to create an environment for me to concentrate on my research.

I would like to express my sincere gratitude to Junpei Mishima, Akinori Nagasaka and Saburo Nakamura as the senior students to provide me kind guidance on the manner and attitude of my research.

I would also extend my thanks to all my friends from the University of Tokyo for providing me a colourful campus life.

Finally, I would express my sincerest gratitude to my families for their confidence in me and unhesitating support to my study on abroad through these years.

References

[1] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," ACM SIGCOMM Computer Communication Review, vol. 32, no. 1, p. 66, 2002.

[2] G. Hampel, M. Steiner, and T. Bu, "Applying software-defined networking to the telecom domain," in 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2013, pp. 133–138.

[3] N. Alliance, "Ngmn 5g white paper," Next generation mobile Networks, white paper, pp. 1–125, 2015.

[4] M. Carugi, "Key features and requirements of 5g/imt-2020 networks," Internetsociety. org, February, 2018.

[5] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," IEEE Communications Magazine, vol. 55, no. 5, pp. 94–100, 2017.

[6] J. Gil Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 518–532, 2016.

[7] L. Gifre, M. Ruiz, and L. Velasco, "Castor: A monitoring and data analytics architecture to support autonomic domain and slice networking," in 2018 20th International Conference on Transparent Optical Networks (ICTON), 2018, pp. 1–4.

[8] G. Sun, Y. Li, D. Liao, and V. Chang, "Service function chain orchestration across multiple domains: A full mesh aggregation approach," IEEE Transactions on Network and Service Management, vol. 15, no. 3, pp. 1175–1191, 2018.

[9] X. An, C. Zhou, R. Trivisonno, R. Guerzoni, A. Kaloxylos, D. Soldani, and A. Hecker, "On end to end network slicing for 5g communication systems," Transactions on Emerging Telecommunications Technologies, vol. 28, no. 4, p. e3058, 2017.

[10] 3GPP, "System architecture for the 5G System (5GS)," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 23.501, 9 2019, version 16.2.0. [Online]. Available: http://www.3gpp.org/DynaReport/23501.htm

[11] A. Wang, M. Iyer, R. Dutta, G. N. Rouskas, and I. Baldine, "Network virtualization: Technologies, perspectives, and frontiers," Journal of Lightwave Technology, vol. 31, no. 4, pp. 523–537, 2013.

[12] Y. Zaki, L. Zhao, C. Goerg, and A. Timm-Giel, "Lte wireless virtualization and spectrum management," in WMNC2010. IEEE, 2010, pp. 1–6.

[13] D. Astely, E. Dahlman, A. Furuskär, Y. Jading, M. Lindström, and S. Parkvall, "Lte: the evolution of mobile broadband," IEEE Communications Magazine, vol. 47, no. 4, pp. 44–51, 2009.

[14] M. Yang, Y. Li, L. Zeng, D. Jin, and L. Su, "Karnaugh-map like online embedding algorithm of wireless virtualization," in The 15th International Symposium on Wireless

Personal Multimedia Communications. IEEE, 2012, pp. 594–598.

[15] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "Nvs: A substrate for virtualizing wireless resources in cellular networks," IEEE/ACM Transactions on Networking, vol. 20, no. 5, pp. 1333–1346, 2012.

[16] G. Bhanage, I. Seskar, R. Mahindra, and D. Raychaudhuri, "Virtual basestation: Architecture for an open shared wimax framework," in Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures, 2010, pp. 1–8.

[17] A. Gonzalez, S. Kuhlmorgen, A. Festag, and G. Fettweis, "Resource allocation for block-based multi-carrier systems considering qos requirements," in GLOBECOM 2017 - 2017 IEEE Global Communications Conference, 2017, pp. 1–7.

[18] A. Aijaz, "<inline-formula><tex-math notation="latex">Hap – SliceR</tex-math></inline-formula>: A radio resource slicing framework for 5g networks with haptic communications," IEEE Systems Journal, vol. 12, no. 3, pp. 2285–2296, 2018.

[19] C. Liang and F. R. Yu, "Distributed resource allocation in virtualized wireless cellular networks based on admm," in 2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2015, pp. 360–365.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, pp. 436–444, 2015.

[21] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., "Human-level control through deep reinforcement learning," nature, vol. 518, no. 7540, pp. 529–533, 2015.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.

[24] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "Nfv and sdn—key technology enablers for 5g networks," IEEE Journal on Selected Areas in Communications, vol. 35, no. 11, pp. 2468–2478, 2017.

[25] S. E. Elayoubi, S. B. Jemaa, Z. Altman, and A. Galindo-Serrano, "5g ran slicing for verticals: Enablers and challenges," IEEE Communications Magazine, vol. 57, no. 1, pp. 28–34, 2019.

[26] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," IEEE Communications surveys & tutorials, vol. 18, no. 1, pp. 236–262, 2015.

[27] ONF, "Software-defined networking: The new form for networks," Open Networking Foundation, Tech. Rep., April 2012. [Online]. Available: https://www.opennetworking.

org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[28] H. Holma and A. Toskala, LTE for UMTS: OFDMA and SC-FDMA based radio access. John Wiley & Sons, 2009.

[29] M. Richart, J. Baliosian, J. Serrat, and J.-L. Gorricho, "Resource slicing in virtual wireless networks: A survey," IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 462–476, 2016.

[30] B. Chazelle, "Efficient implementation," IEEE Transactions on Computers, vol. 32, no. 8, 1983.

[31] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in Proceedings of the AAAI conference on artificial intelligence, vol. 30, no. 1, 2016.

[32] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in International conference on machine learning. PMLR, 2016, pp. 1995–2003.

Publications

[1] H. Zhou, H. Aida, "Network Slicing Resource Allocation Algorithm Based on Deep Q-Learning," Technical Committee on Communication Quality (CQ), The Institute of Electronics, Information and Communication Engineers (IEICE), September 2021.