A Path Forward: 6G Resource Allocation from a Deep Q-Learning Perspective

Lucas Inglés, Claudina Rattaro and Pablo Belzarena
Instituto de Ingeniería Elécrica, Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
{lucasi,crattaro,belza}@fing.edu.uy

***Abstract*—The 6G paradigm presents a myriad of challenges, as it promises complex features such as managing diverse traffic profiles under a unified infrastructure. While many studies propose deep-Q learning (DQN) approaches for resource management in Network Slicing (NS) schemes, these algorithms often face a core issue: they are not easily reproducible in real-world environments due to their high dimensionality. In this study, we analyze a distributed DQN-based radio resource allocation methodology, designed to efficiently meet specific Service Level Agreements (SLAs). Our contribution includes making the code publicly available for further research and evaluation. We then assess its performance through a comparison with a Baseline DQN approach, highlighting the strengths and limitations of both models.**

***Index Terms*—6G, Resource Allocation, Deep Q-Learning, Network Slicing**

I. Introduction

In a world where the speed and connectivity of 5G have transformed our expectations, the transition to 6G networks demands even more advanced and flexible infrastructures to accommodate the increasing complexity of communication systems. A key innovation in this area is **Network Slicing (NS)**, which allows a single network to be segmented into multiple virtual slices, each tailored to specific service requirements.

However, in 6G, NS presents unique challenges due to the diversity of services and the need for highly customized functionalities. Efficient resource management is critical to ensure that **Service Level Agreements (SLAs)** are met. This necessitates the use of flexible, self-optimizing systems capable of adapting in real-time to changing network conditions.

Deep Reinforcement Learning (DRL), and particularly **Deep Q-Learning (DQN)**, has emerged as a promising approach to addressing these challenges [1] [2]. DQN enables the optimization of network resources in real time, dynamically adapting to meet the stringent demands of 6G networks, such as increased data traffic, ultra-low latency, and enhanced **Quality of Service (QoS)** and **Quality of Experience (QoE)** requirements.

However, the application of DQN in 6G network slicing is not without its difficulties. The intricate dynamics of 6G networks, coupled with their complex service-level requirements, pose significant challenges for real-time resource allocation.

In this research, we explore the use of DQN for resource allocation in 6G network slices [3], building on prior work [4], and provide a comparative analysis with a baseline DQN approach to assess performance improvements.

The remainder of this paper is organized as follows: **Section II** discusses the theoretical background and the specific challenges of network slicing in 6G. **Section III** details the DQN algorithm and its implementation. **Section IV** presents the results of our comparative analysis. Finally, **Section V** concludes with a summary of our findings and future directions.

II. Fundamentals

A. Radio resource allocation challenges in 6G

Efficient resource allocation optimizes system performance, ensures fairness, and meets diverse QoS requirements in 6G. In 5G, Orthogonal Frequency Division Multiplexing (OFDM) is used, and it is expected that 6G will follow a similar path while still considering the evaluation of other options. The OFDM system enhances data transmission by partitioning the frequency spectrum into multiple orthogonal subcarriers. The frequency domain is segmented into Resource Blocks (RBs), each comprising 12 subcarriers. RBs are the minimal units for resource allocation, forming a grid in both time and frequency dimensions.

RB allocation directly impacts on the user QoE: more resources enable higher data exchange and lower communication delays. However, careful allocation is necessary for optimal performance. 6G networks face several challenges: **Heterogeneous Integration**, merging diverse technologies while maintaining service delivery; **Surging Data Traffic**, managing increased connectivity demands; **Network Density**, addressing complexities in ultra-dense deployments; and **Dynamic Environments**, adapting to fluctuating user mobility and varying conditions.

CSIC R&D project: 5/6G Optical Network Convergence: an holistic view

Artificial Intelligence (AI) and Network Slicing offer advanced solutions for these challenges.

B. Network Slicing and Dynamic Resource Allocation

Network slicing (NS) allows subdividing a unified network infrastructure into distinct segments or "slices" for different services, ensuring dedicated QoE levels. As depicted in Fig. 1, two levels of scheduling are required: intra-slice and inter-slice.

Inter-Slice Scheduling: Prioritizes resource allocation among slices according to strategic objectives and service hierarchies to meet SLAs.

Intra-Slice Scheduling: Manages resource distribution within each slice, ensuring provisioning without affecting other slices.

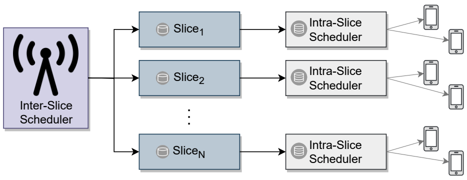


Fig. 1: Inter and Intra slice scheduling scheme.

This research focuses on inter-slice resource allocation because of its broader impact on overall network efficiency and the ability to manage diverse Service Level Agreements (SLAs). By prioritizing resources between different services, inter-slice scheduling optimizes network performance, making it a key factor for ensuring SLA compliance.

C. Reinforcement Learning and Deep Q-Learning

Reinforcement Learning (RL) enhances decision-making through rewards and penalties, with DQN being an advanced form of RL. At the core of DQN are Q values, which represent the future expected rewards of actions taken in specific states. By iteratively updating these Q values based on previous outcomes, the algorithm gradually learns an optimal policy for resource allocation.

The Q value is calculated using the Bellman equation, which assesses the cumulative reward of an action, accounting for both immediate and future rewards. This allows DQN to make decisions that optimize long-term performance, making it particularly suitable for complex, dynamic environments such as 6G network slicing.

More specifically, DQN can optimize dynamic resource allocation within Radio **Access Networks (RAN)**, improving both QoE and overall network performance. Its efficient Q-value updates, off-policy flexibility, and model-free learning make it ideal for addressing complex state spaces, such as those in mobile networks.

However, selecting appropriate state and action policies is crucial to avoid impractical solutions. Our research focuses on identifying effective DQN strategies for resource allocation in mobile networks, ensuring the algorithm's suitability for real-world deployment.

D. Simulator Framework

Given the ongoing development of 6G, we use the *Py5cheSim* 5G network simulator to evaluate the DQN algorithm. *Py5cheSim* [5] supports NS and variable numerology, offering a comprehensive environment for testing and real-world application analysis, providing insights into DQN's operational challenges and benefits in 5G/B5G networks.

III. Scheduling algorithms

This study analyzes inter-slice resource allocation in 6G networks using DQN algorithms. Existing DQN-based research primarily focuses on centralized solutions, which often face challenges with dimensionality, learning efficiency, and computational processing. The work in [4] introduces a Distributed Scheduler for resource management, which we adopt in our research.

To evaluate the efficacy of the distributed scheduler, we compare it with a Baseline centralized algorithm that offers a holistic system perspective. This comparison helps highlight the strengths and weaknesses of the Distributed Scheduler.

A. Baseline Scheduler

Inspired by [6], the Baseline Scheduler operates within a non-distributional framework, assessing the overall system state and implementing actions that influence the entire system. Rewards are derived from system performance metrics. The state, action, and reward definitions are as follows:

1) State: The state is defined as the number of packets received by each slice during a time window, represented as a vector of n elements. Each element corresponds to the state of a slice, influenced by the traffic profile. Higher traffic results in larger numerical values.

2) Action: Actions allocate resources to slices by selecting from a set of predefined allocations. For example, with two slices and four RBs, possible allocations are [1, 3], [2, 2], or [0, 4]. This approach balances control precision with efficient learning.

3) Reward: The reward function aims to minimize resource waste and ensure that sufficient resources are available for each slice to meet the SLAs. Rewards are calculated as the aggregate Resource Block Usage Ratio (RBUR) across slices and the number of users meeting QoE benchmarks. Constants ψ and β balance the importance of RBUR and QoE, both set to 1 in this study.

B. Distributed Scheduler

In the distributed model, individual network slices independently report their status to the base station at time t. The base station calculates the reward for previous actions taken at $t-1$ and stores this in the experience database. Based on each slice's current state and integrated model guidelines, the Resource Blocks (RBs) are allocated accordingly. This approach ensures that each slice receives resources in a way that meets its specific SLA requirements. This behavior is depicted in Fig. 2.

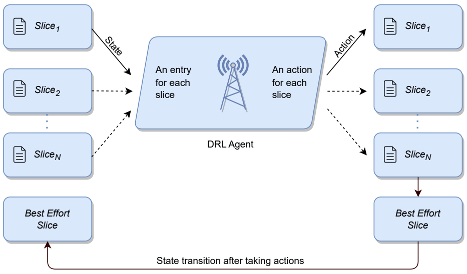


Fig. 2: Distributed scheduler model.

The distributed scheduling approach used in this paper is adopted from the methodology proposed by Abiko et al. [4], where a distributed mechanism was introduced for resource management in network slicing. In our work, we have reproduced a simplified version of their algorithm and made its implementation publicly available. Additionally, we have reduced the dimensionality of the state representation, we have used a different simulator, and we have compared its performance against a Baseline centralized algorithm in various traffic conditions to assess scalability and efficiency from an SLA compliance perspective. Our implementation and comparative analysis provide deeper insights into the practical challenges and benefits of distributed scheduling for 6G networks.

1) State: The state is defined by a six-element tuple: **Network Slice Requirements Satisfaction (NSRS)**, **Ratio of assigned and used Resource Blocks (RBUR)**, Number of assigned Resource Blocks, Throughput requirement, Number of **User Equipments (UEs)** within the slice, and Accumulated traffic in the buffer.

NSRS indicates user satisfaction with throughput demands; values near one indicate high satisfaction, while values near zero indicate unmet service levels. RBUR measures resource utilization; values near one signify optimal usage, while values near zero indicate overallocation. The number of RBs allocated at time t is the third element.

Throughput characterizes each slice, although other QoS metrics such as delay and jitter may be relevant in future studies. The demand level and accumulated traffic provide insight into the slice's condition and rewards.

Incorporating multiple attributes enhances precision but increases model complexity and learning cost. The state configuration was modified from the original work [4] to fit research objectives and simulator conditions, aligning evaluation with the original study's outcomes.

2) Action: In this model, an action adjusts resource allocation for a slice, quantified as a change in RBs using **Resource Block Adjustment (RBA)**[1]:

$$RBA = \left\lfloor (-1)^a \times 2^{\lfloor a/2 \rfloor - 1} \right\rfloor, \tag{1}$$

where a is a discrete output from the decision-making algorithm. The scope of actions is restricted to eight to maintain stability and avoid abrupt changes. Equation (1) uses a floor function to ensure integer results.

The current allocation of RBs at time t (ARB_t) is:

$$ARB_t = ARB_{t-1} + RBA. \tag{2}$$

This model adjusts resources for individual slices without considering the states of other slices, enhancing focus and efficiency. A "Best Effort" slice acts as a dynamic reserve, accommodating varying demands from non-contractual users based on network conditions, absorbing unused RBs from other slices.

3) Reward: The reward is defined as:

$$R = NSRS \cdot RBUR, \tag{3}$$

where NSRS and RBUR balance resource allocation and user requirement fulfillment. The product of these indicators influences decision evaluation, aiming to maximize both. NSRS and RBUR range from 0 to 1, so R also falls within this range. An R value of 1 indicates optimal resource allocation and requirement satisfaction.

IV. Evaluation

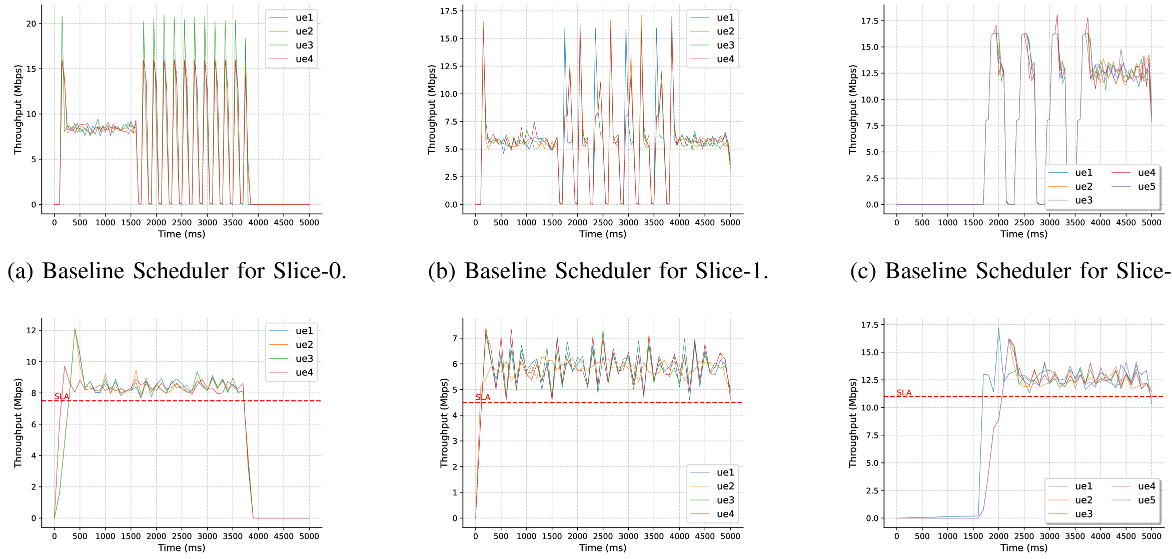
Using the Py5cheSim simulation framework, we conducted an empirical analysis of both algorithms under the specified traffic conditions. The implementations, available at [7], were developed using neural network models built with the Keras library in Python. The following sections describe the common evaluation scenario for both algorithms, present the outcomes, and provide a comprehensive analysis.

A. Traffic profiles

For the evaluation, we tested three groups: Slice-0, Slice-1, and Slice-2, each with different transmission parameters (see Table I). The objective was to assess both algorithms' performance with varying traffic profiles. The SLA values were based on optimal functioning without RB restrictions.

To assess the impact of varying active slices, we configured slices with different activation patterns:

[1] In the work of [4] this quantity is called IDRB



(a) Baseline Scheduler for Slice-0.

(b) Baseline Scheduler for Slice-1.

(c) Baseline Scheduler for Slice-2.

(d) Distributed Scheduler for Slice-0.

(e) Distributed Scheduler for Slice-1.

(f) Distributed Scheduler for Slice-2.

Fig. 3: Results of user throughput over time for both algorithms.

TABLE I: Evaluation traffic profiles

	Traffic parameters		
	Slice-0	***Slice-1***	***Slice-2***
# clients	4	4	5
Pkt. size	300 bytes constant	Pareto [Mean = 410 bytes]	Lognormal [Mean = 800 bytes]
Pkt. arrival	Uniform [0, 0.6] ms	Uniform [0, 1.2] ms	Uniform [0, 1] ms
SLA	7.5 Mbps	4.5 Mbps	11 Mbps

Slice-0 is active from 0% to 75% of simulation time, Slice-1 is active throughout 100% of the simulation, and Slice-2 is active from 25% to 100% of simulation time.

B. Results and analysis

Both algorithms were trained using the same equipment and simulation settings. The results are shown in Figs. 3(a,b,c) show the results for the baseline scheduler, while Figs. 3(d,e,f) display the results for the distributed approach. The simulation lasted a total of 5 seconds, allowing for a more precise examination of the effects of resource allocation, as inter-slice assignments were made every 100 milliseconds.

1) Baseline: The results show that when only two out of three slices are active, i.e., in the ranges [0%,25%] and [75%,100%] of the simulation time, the network performs as desired, with sufficient RBs to meet each user's needs and the SLA.

When all slices are active, i.e., in the range [25%,75%], the system performs poorly, characterized by intermittent disruptions in user throughput. The scheduler misprioritizes slices, leaving some with insufficient resources and reducing network availability.

The state definition does not accurately reflect the system and lacks a strong correlation with action and reward. It prioritizes emptying the packet buffer of the slice with the most packets, neglecting slices with fewer packets.

The states and actions lack granularity. Refining them extensively is prohibitive due to memory requirements, leading to imprecise assignments and overprovisioning of resources, impairing reward and learning.

When analyzing RBUR, we operated in a scenario with enough RBs to meet all slice requirements, leading to overassignment. Incorporating RBUR into the reward function poses challenges, suggesting that weighting the QoE reward higher might be more effective.

The DQN solution encounters difficulties in complex scenarios. The current prioritization strategy, based on packet queue size, struggles to achieve optimal resource allocation and desired QoE. Handling RBUR is intricate, and scaling the problem to more slices increases complexity, making this solution less scalable.

Thus, a basic model is unsuitable for this problem. It overlooks much information, and decisions based on an inaccurate buffer representation lead to inefficiency. While it may work in specific situations, extensive training and practical considerations beyond this research's scope make the model inefficient.

2) Distributed Scheduler Solution: Fig.4 shows the inter-slice resource allocation. The three simulation stages, during which different slices are activated, are distinguishable. The smooth slope of resource allocation during slice activation and deactivation indicates granular resource growth without abrupt jumps. Resources align with expected service levels, with Slice-2 receiving the most resources. The Best Effort slice acts as a buffer, providing resources to other slices as needed.

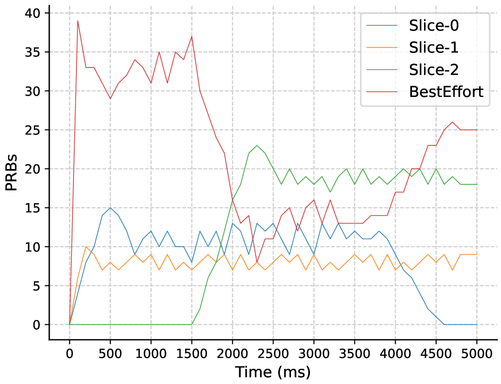


Fig. 4: Resources allocated in the Distributed Scheduler.

Figs. 3(d,e,f) show the throughput of users for Slice-0, Slice-1, and Slice-2, respectively, over simulation time. Each figure includes a horizontal line representing the SLA for the respective slice. Performance meets SLA requirements at all times, regardless of the number of active slices.

The comprehensive definition of action and state sets accurately characterizes the network and strongly correlates with the reward. The algorithm's learning time is shorter than the Baseline Scheduler due to distributed learning, which stores possible states for a single slice rather than all possible combinations. This reduces learning time, crucial for real network adoption.

Resource utilization for Slice-1 (the most unpredictable traffic) exceeds 81%, and is even higher for other slices.

The results demonstrate that the distributed scheduler performs efficiently in the simulated environment, achieving better resource allocation than the baseline. However, the convergence time of the model is closely related to the traffic profile, and the ability to adapt to changing traffic patterns may require further adjustments. Although the algorithm demonstrates robustness in stable traffic scenarios, there may be a need to explore adaptive mechanisms that allow the model to adjust more quickly to sudden shifts in traffic demand without requiring retraining. Addressing these limitations could further enhance the flexibility and responsiveness of the system.

V. Conclusions and Future Work

In summary, the distributed scheduler outperformed the non-distributed one in SLA compliance, resource utilization, learning time, and service availability. A key contribution of this work is making the code and simulator publicly available, enabling further comparative analyses and improvements.

Exploring more sophisticated action selection mechanisms can optimize resource allocation and balance slices better. Addressing issues related to model definition, state and action granularity, and precise resource assignment is crucial for desired network behavior and optimization in multi-slice environments, achievable through distributed learning.

The Distributed approach shows promise for implementing DQN-based algorithms in 6G, offering flexibility and resource optimization. It performs well for predictable traffic slices, and even with variability, resource allocation meets slice requirements, but can improve in RBUR.

Implementing DQN in a real network requires careful consideration due to its impact on user performance. Techniques and tools are needed to mitigate low network availability.

Although the algorithm performs well in the current setup with a limited number of clients, scalability remains a critical concern as the number of clients increases. In future work, we aim to study the impact of DQN algorithms in larger network scenarios with a higher number of users. Distributed learning techniques are expected to enhance the algorithm's scalability by reducing the problem's dimensionality. As network complexity grows, it is essential to ensure that the DQN algorithm remains both efficient and effective.

References

[1] W. Guan, H. Zhang, and V. C. Leung, "Customized slicing for 6g: Enforcing artificial intelligence on resource management," *IEEE network*, vol. 35, no. 5, pp. 264–271, 2021.

[2] J. A. Hurtado Sánchez, K. Casilimas, and O. M. Caicedo Rendon, "Deep reinforcement learning for resource management on network slicing: A survey," *Sensors*, vol. 22, no. 8, p. 3031, 2022.

[3] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6g wireless communication systems: Applications, requirements, technologies, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 957–975, 2020.

[4] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, and H. Mineno, "Radio resource allocation method for network slicing using deep reinforcement learning," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 420–425.

[5] G. Pereyra, C. Rattaro, and P. Belzarena, "Py5chesim: a 5g multi-slice cell capacity simulator," in *2021 XLVII Latin American Computing Conference (CLEI)*, 2021, pp. 1–8.

[6] R. Li, Z. Zhao, Q. Sun, I. Chih-Lin, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74 429–74 441, 2018.

[7] "Py5chesim simulator with the implemented algorithms," https://github.com/lingleslоggia/py5chesim.