Reinforcement Learning Scheduling for URLLC Service Protection in Industry 4.0 Scenario

Cleverson Nahum, Weskley V. F. Mauricio, Maykon Silva, Michelle S. P. Facina, Marcos Y. Takeda and Aldebaro Klautau

***Abstract*— In this work, we study the inter- and intra-slice scheduling of the Industry 4.0 scenario considering three services available: enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), and Massive Machine Type Communication (mMTC). In this context, we proposed inter-slice scheduling with Ultra-Reliable Low-Latency Communication (URLLC) service protection using the reinforcement learning agent. In our results, to get clos er to a practical Industry 4.0 scenario, we utilized channel measurements of an actual factory hall in Nuremberg, Germany. The superiority of the proposed framework is demonstrated through numerical simulations compared with reference solutions.**

***Keywords*— Resource Scheduling, Industry 4.0, Slices, Reinforcement Learning, Service Protection.**

I. Introduction

The 5th Generation (5G) of mobile networks has played a significant role in developing cellular systems, causing them to become increasingly widespread in industrial and agricultural automation. Thanks to them, it is now possible to interconnect many intelligent devices such as sensors, drones, planes, and vehicles [1]. However, several heterogeneous use cases also surfaced related to different application areas. As an example, use cases supported by 5G related to Industry 4.0 have been defined by the 5G Alliance for Connected Industries and Automation (5G-ACIA), and 3rd Generation Partnership Project (3GPP) have different communication requirements [2]. Various levels of Quality of Service (QoS) in data rates, delay, reliability, and availability have been established.

Therefore, it is interesting to make future generations of cellular networks flexible and adaptable to different applications and user requirements. For example, 5G networks have already introduced significant innovations to support the digitization of verticals, such as a New Radio (NR) interface with different numerology for the flexible use of radio resources. Already network slicing is a promising paradigm that exploits virtualization and networking software to create different instances of independent logical networks on a common physical network infrastructure in both the Radio Access Networks (RANs) and the network core. Each instance is tailored for specific QoS profiles, so network slicing can simultaneously support multiple mobile services. Basically, a slice is a tailored isolated End-to-End (E2E) that contains resources from various network domains and can accommodate a specific service.

Cleverson Nahum, Marcos Y. Takeda and Aldebaro Klautau are part of Federal University of Pará, Weskley V. F. Mauricio, Maykon Silva, and Michelle S. P. Facina are part of Fundação Centro de Pesquisa e Desenvolvimento em Telecomunicações (CPQD), Campinas-SP, Brazil. e-mail: {cleverson, aldebaro}@ufpa.br,{wmauricio, mpsilva, mfacina}@cpqd.com.br.

However, a vital issue in RAN slicing is Radio Resource Scheduling (RRS), which is responsible for adequately allocating the limited number of Resource Blocks (RBs) (radio resources) to different individual users with different QoS according to traffic variations and the dynamic state of the channel. Unfortunately, RAN resource scheduling is challenging due to radio channel variations, performance isolation, diversified service requirements, and user mobility [3].

In traditional systems, resource scheduling allows a rigid way of exploiting resources among users to achieve high-gain multiplexing with spectrum sharing [4]. Currently, the main goal of resource scheduling is to flexibly and adaptively share resources among slice owners so that the infrastructure can be utilized efficiently. During this process, it is necessary to maintain a certain degree of slice independence (performance and functional isolations) so that the tenants can maintain complete control of their custom slices to meet your service needs. When focusing on Industry 4.0 scenarios, the protection of critical applications (usually associated with URLLC slices) is an essential demand for RRS due to the network and device's channel variations that could cause instability. It requires constant adaptations from the RRS to still fulfill the critical application requirements.

Following this reasoning, our contributions are: (i) the development of a dataset and a simulator[1] for Industry 4.0 scenario in Python; (ii) a study of URLLC service protection using the reinforcement learning agent as resource scheduling. To emulate a more realistic Industry 4.0 environment, we have integrated a traffic and channel simulator developed in Python programming language and Quasi Deterministic Radio Channel Generator (QuaDRiGa), considering eMBB, mMTC, and URLLC. As the results of [5] suggest, the use of the 3GPP and QuaDRiGa channel models for industrial systems in a frequency range from 2 GHz to 6 GHz enable more accurate simulation studies for 5G and beyond-5G wireless communication system. It is worth noting that the indoor factory has many metal machine tools, making its radio propagation characteristics and corresponding channel models significantly different from those of the indoor office and indoor hotspot [6]. Then, using conventional optimization techniques to solve the prediction and online resource scheduling problems would be unfeasible.

II. System model

This section describes the scenario of this work, with channel parameters, RAN slicing scheme, and traffic

[1] https://github.com/lasseufpa/rrs_industrial_scenario

model.

A. Channel

We consider the downlink of an Frequency Division Duplex (FDD) Orthogonal Frequency Division Multiple Access (OFDMA) industrial indoor system, such as a factory hall, composed of a single antenna Base Station (BS). The system has U User Equipments (UEs) equipped with omnidirectional single-antennas. We consider that there are three slice types with specific use cases, required delay, and achievable data rate. For simplicity, we assume that each UE belongs to only one slice type. We adopt that each Transmission Time Interval (TTI) has R RBs, in which the RB is the smallest resource unit that can be allocated to a UE. Furthermore each RB is composed of N_{sc} adjacent OFDMA subcarriers and N_{symb} consecutive Orthogonal Frequency Division Multiplexing (OFDM) symbols.

Considering that the BS allocates the r-th RB to u-th UE, the perceived Signal to Noise Ratio (SNR) by the UE is given by

$$\gamma_{u,r} = \frac{\alpha_j \; p_r \; |h_{u,r}|^2}{\sigma_u^2}, \tag{1}$$

where $h_{u,r}$ is the downlink channel of u-th UE with allocated r RB, p_r is the allocated transmit power to the r-th RB, α_u is the effect of path gain and shadowing experienced by the u-th UE and σ_u is the noise power experienced by the u-th UE. Therefore, we define the spectral efficiency $\mathsf{S}_{u,n}(n)$ to RB r and UE u as

$$\mathsf{S}_{u,r}(n) = \log_2(1 + \gamma_{u,r}). \tag{2}$$

In this paper, we used the QuaDRiGa Industrial Indoor Scenario with Line Of Sight (LOS) [7]. This scenario was prepared for industrial-indoor deployments, such as factory halls, in Industry 4.0 environments. Also, it covers carrier frequencies from 2 GHz to 6 GHz. The detailed description of the channel parameters used to generate the scenario, such as shadow fading, small fading, angular spread, and the number of clusters, are found in [5], [7].

B. RB allocation and RAN slicing

When considering the RRS in a scenario with RAN slicing, the scheduling process is usually divided between two different schedulers called inter- and intra-slice scheduling. The former distributes the BS's RBs available among the slices, while the latter distributes the RBs received from the inter-slice scheduling among the UEs associated with the slice [8]. The main difference compared to the scenario without slices is the focus on meeting the needs of a group of UE instead of meeting the needs of each UE individually [9]. Therefore, the purpose of the scheduler in this scenario is to meet the requirements of each network slice.

Each slice s contains a set of U_s UEs with similar traffic behavior and the same QoS requirements. A vector

$$\boldsymbol{R}_n = [R_1(n), R_2(n), \ldots, R_S(n)], \tag{3}$$

defines the number of Resource Block Groups (RBGs) allocated for each slice by the inter-slice scheduling in the simulation step n, where $R_s(n)$ represents the number of RBGs allocated to slice s at step n. The RRS process obeys to

$$\sum_{s=1}^{S} R_s(n) = R, \tag{4}$$

where the sum of all RBGs distributed along with the slices is always equal to the total amount of RBGs available R. Therefore, the main function of the RRS in a scenario with RAN slicing is to define $R_s(n)$ for each slice s in a step n in accordance with the network conditions to satisfy the slice requirements. Once the RBs are distributed among the slices, the intra-slice scheduling is responsible for distributing each among the slice's UEs.

The slice requirements define the target values for each monitored network metric. We define the slice requirements considering three main metrics: served throughput, buffer occupancy, and buffer delay. The served throughput for each UE is the maximum rate in bits per step that an UE can obtain, considering the number of RBG allocated to it and its spectral efficiency. In other words,

$$r_u(n) = \left\lfloor \frac{(R_s^u(n)/R)\text{BS}_u(n)}{P} \right\rfloor P, \tag{5}$$

where $R_s^u(n)$ represents the number of RBG allocated to u-th UE by intra-slice scheduling, B is the total bandwidth available in the BS, $S_u(n)$ is the spectral efficiency for u-th UE at step n, and P is the packet size.

The buffer occupancy rate is defined as

$$b_u^{\text{occ}}(n) = \frac{b_u(n)}{b_{\max}}, \tag{6}$$

where $b_u(n)$ represents the amount of data available in the buffer of u-th UE in the simulation step n and $b_{\max}$ is the maximum buffer capacity of UE. Packets are dropped every time the buffer is full or the delay of a packet exceeds the maximum delay $d_{\max}$ allowed by the buffer.

The buffer delay represents the average time each packet waited before being sent or lost and is defined as

$$d_u(n) = \frac{\sum_{i=0}^{d_{\max}} id_n^u(i)}{\sum_{i=0}^{d_{\max}} d_n^u(i)}, \tag{7}$$

where $d_u(n)$ is a vector of dimension $d_{\max} + 1$ representing the delay of packets in the buffer of u-th UE at step n.

C. Slice Types and Requirements

We define network slice metrics as an average of the UEs metrics associated with a specific slice s. So, a specific metric SM_s for slice s is

$$\mathsf{SM}_s = \frac{\sum_{u=1}^{U_s} \mathsf{SM}_u}{U_s}, \tag{8}$$

where SM_u can represent the served throughput, buffer occupancy, and buffer delay of a specific UE u. U_s represents the total number of UEs associated with slice s. Each slice type has different requirements. This work assumes three different slice types: eMBB, URLLC, and mMTC.

1) eMBB slice: The UEs assigned to the eMBB slice require high throughput, regardless of the channel conditions, and they do not have stringent delay and packet loss requirements. Augmented reality applications to assist industry analysis is an example of an eMBB application. We defined two main requirements for the eMBB slice: the served throughput $r_{\text{embb}}(n)$ should be equal to or above a specified minimum served throughput $r_{\text{embb}}^{\text{req}}$. The buffer delay $d_{\text{embb}}(n)$ should be kept below a required buffer delay $d_{\text{embb}}^{\text{req}}$. The requested throughput from UEs u associated with eMBB slice $r_{\text{u}}^{\text{rqt}}(n)$ is a Poisson distribution with mean μ_{embb} [10].

2) URLLC slice: The UEs assigned to a URLLC slice requires low latency and ultra-reliable communication, usually characterized by a low packet loss rate. Some examples of URLLC applications are monitoring an automation process and its remote control, which does not require a great throughput but requires high communication reliability and low latency to work well. URLLC slice usually serves critical applications. Therefore, the network must prioritize the URLLC slice and avoid violating its slice requirements. The URLLC evaluated metrics are the same from eMBB slice URLLC, with $r_{\text{urllc}}(n)$ and $d_{\text{urllc}}(n)$ representing the URLLC served throughput and buffer delay. With $r_{\text{urllc}}^{\text{req}}$, and $d_{\text{urllc}}^{\text{req}}$ representing the URLLC intents for served throughput and buffer delay, respectevely. The requested throughput $r_{\text{u}}^{\text{rqt}}(n)$ from URLLC UEs u is defined as a Poisson distribution with mean μ_{urllc}.

3) mMTC slice: The mMTC slice is used to connect many devices. Internet of Things (IoT) devices, such as sensors with intermittent traffic, are essential examples of mMTC applications in industrial scenarios. We assume that the mMTC considers that the buffer delay $d_{\text{urllc}}(n)$ should be equal to or below a required buffer delay $d_{\text{mmtc}}^{\text{req}}$. The mMTC UEs are activated or deactivated in each step with a 50% probability. The requested throughput for UE u from mMTC slice $r_{\text{u}}^{\text{rqt}}(n)$ is defined as a Poisson distribution with mean μ_{mmtc} if the UE is activated or zero otherwise.

Due to UEs channel variations, the network radio resources may not be sufficient to satisfy all the slice requirements during the entire simulation. Therefore, it is essential to ensure that the URLLC applications have their requirements prioritized in relation to the other slices due to their critical applications.

III. Proposed RRS agent

This work proposes an Reinforcement Learning (RL) agent to perform inter-slice RRS operations jointly with a round-robin scheduler performing intra-slice RRS operations. We adopt the Soft Actor-Critic (SAC) RL method defined in [11], which optimizes a stochastic policy using an off-policy technique, forming a bridge between stochastic policy optimization and Deep Deterministic Policy Gradient (DDPG) approaches [12]. The SAC method improves the exploration and solves the stability issues presented by off-policy methods. Standard RL methods maximize the expected sum of rewards. However, the technique SAC considers a more general maximum entropy objective which favors stochastic policies by augmenting the objective with the expected entropy.

A. Observation Space

We define the observation space $\boldsymbol{O}_n$ in a given step n as a representation of the state s_t containing knowledgeable information. The observation space is defined as

$$\boldsymbol{O}_n = [\boldsymbol{s}_{\text{embb}}, \boldsymbol{s}_{\text{urllc}}, \boldsymbol{s}_{\text{mmtc}}], \tag{9}$$

where slice metric vectors $\boldsymbol{s}_s = [r_s(n), b_u^{\text{occ}}(n), d_s(n)]$, being composed by the three slice metrics: served throughput, buffer occupancy, and buffer delay, respectively. These slice metrics are the average values obtained from the UEs connected to the analyzed slice.

B. Action Space

We define an action as a vector $\boldsymbol{A}_n$ in a given step n, that is defined as $\boldsymbol{A}_n = [a_{\text{embb}}, a_{\text{urllc}}, a_{\text{mmtc}}]$, where a_s represents an action factor for slice s with value in a range $[-1, 1]$ to match the output of the Gaussian distribution for continuous actions used, improving the learning process [11]. After that, the agent's chosen action $\boldsymbol{A}_n$ is converted to the number of RBGs for each slice

$$\boldsymbol{A}_n^{\text{RB}} = \left\lfloor \frac{R(\boldsymbol{A}_n + 1)}{\sum_{i \in S}(a_i + 1)} \right\rceil. \tag{10}$$

Finally, the inter-slice scheduling decision is applied in the intra-slice scheduling using round-robin to distribute the RBs among the UEs.

C. Reward Calculation

The reward function $\mathsf{W}(n)$ considers the slice requirements as a basis to define how close the slice metrics are to fulfill their intents. So, the reward has one component for each slice type as defined in

$$\mathsf{W}(n) = \begin{cases} \mathsf{W}_{\text{embb}}(n) + \mathsf{W}_{\text{mmtc}}(n), \text{if } \mathsf{W}_{\text{urllc}}(n) < 0 \\ -(3 - 10\mathsf{W}_{\text{urllc}}(n)), \text{Otherwise} \end{cases}, \tag{11}$$

where the $\mathsf{W}_{\text{embb}}(n)$, $\mathsf{W}_{\text{urllc}}(n)$ and $\mathsf{W}_{\text{mmtc}}(n)$ represent the reward for eMBB, URLLC, and mMTC slices, respectively, at step n.

The slice reward calculations for eMBB, URLLC, and mMTC are based on served throughput rate and delay requirements as defined in

$$\mathsf{W}_{\text{embb}}(n) = -(\mathsf{W}_{\text{embb}}^{r}(n) + \mathsf{W}_{\text{embb}}^{d}(n)), \tag{12}$$

$$\mathsf{W}_{\text{urllc}}(n) = -(\mathsf{W}_{\text{urllc}}^{r}(n) + \mathsf{W}_{\text{urllc}}^{d}(n)), \tag{13}$$

$$\mathsf{W}_{\text{mmtc}}(n) = -\mathsf{W}_{\text{mmtc}}^{d}(n), \tag{14}$$

where $\mathsf{W}_{s \in S}^{r}(n)$ and $\mathsf{W}_{s \in S}^{d}(n))$ represent the served throughput and delay contributions to the rewards. The served throughput contribution is defined as:

$$\mathsf{W}_{s}^{r}(n) = \begin{cases} \frac{r_{\text{s}}^{\text{req}} - r_{\text{s}}(n)}{r_{\text{s}}^{\text{req}}}, \text{ if } r_{\text{s}}(n) < r_{\text{s}}^{\text{req}} \\ 0, \qquad \text{if } r_{\text{s}}(n) \geq r_{\text{s}}^{\text{req}} \end{cases}, \tag{15}$$

and the buffer delay contributions is

$$\mathsf{W}_{s}^{d}(n) = \begin{cases} \frac{d_{\text{s}}(n) - d_{\text{s}}^{\text{req}}}{d_{\max} - d_{\text{s}}^{\text{req}}}, \text{ if } d_{\text{s}}(n) > d_{\text{s}}^{\text{req}} \\ 0, \qquad \text{if } d_{\text{s}}(n) \leq d_{\text{s}}^{\text{req}} \end{cases}. \tag{16}$$

The objective of the proposed agent is to maximize the reward function values, maximizing the fulfillment

of the defined requirements. Moreover, it also provides a prioritization of the URLLC slice in the Equation 11 where in case the URLLC slice requirements are not satisfied, it receives a negative reward equivalent to the unfulfillment of eMBB and mMTC slices plus the distance to fulfill the URLLC slice multiplied by ten. Therefore, the proposed agent is always incentivized to satisfy the URLLC requirements first.

D. Baseline Agent

We utilize a baseline for comparison with our proposed method, where the baseline uses the same observation and action space from our proposed method. It utilizes a reward calculation based on a modified SLA Satisfaction Rate (SSR) method [13]. The reward function is defined as

$$\mathsf{W}(n) = \mathsf{W}_{\text{embb}}(n) + \mathsf{W}_{\text{mmtc}}(n) + \mathsf{W}_{\text{urllc}}(n). \quad (17)$$

IV. Numerical Results

Table I presents the communication network parameters used in the channel simulation. We utilize an RL agent to implement the inter-slice scheduling and a round-robin method to perform the intra-slice scheduling.

TABLE I: Simulation Parameters.

Parameter	**Value**
Bandwidth	100 MHz
Carrier frequency	10 GHz
Number of RBs	100
Subcarrier spacing	15 kHz
TTI duration	1 ms
Transmit power	35 dBm
UEs Speed	3 km/h
Number of UEs	100
Number of slices	3
Simulation rounds	200
TTI	1000
Throughput req.	$r^{\text{req}}_{\text{embb}}$=20 and $r^{\text{req}}_{\text{embb}}$=5 Mbps
Latency req.	$d^{\text{req}}_{\text{embb}}$ = 30, $d^{\text{req}}_{\text{urllc}}$ = 1, and $d^{\text{req}}_{\text{mmtc}}$ = 50 ms
Requested thr.	μ_{embb} = 20, μ_{urllc} = 5, and μ_{mmtc} = 0.1 Mbps

The scenario considered is shown in Fig. 1, where a single antenna BS is positioned at the top corner of a factory hall serving UEs with three different use cases corresponding to the eMBB, URLLC and mMTC applications explained in the Sub-section II-C. To make the scenario closer to reality, we have extracted the factory hall measurements from [7], which are measurements of an actual factory hall in Nuremberg, Germany. The BS serves 100 UEs, which are uniformly distributed inside of the factory hall, being 20 UEs associated to eMBB, 20 UEs associated to URLLC, and 60 UEs associated to mMTC applications. The industrial channel model contains 100 RBs to be distributed among slices and UEs using the inter- and intra-slice scheduling and an equally divided power allocation among RBs. Furthermore, the UEs are moving at 3 km/h and the system has 3 slices (1 for eMBB, 1 for URLLC, and 1 for mMTC).

We divided the 200 simulation rounds generated by the QuaDRiGa simulation into a training and testing set, in which the former contains 160 rounds, and the latter includes 40 rounds, totalizing 160000 and 40000 simulations

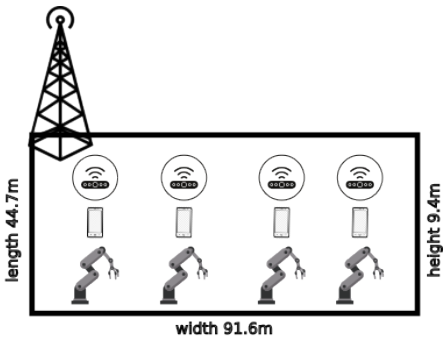

Fig. 1: Scenario with a BS at left top corner, three use cases UEs (sensor, mobile phone, and robotic arm) inside a factory hall.

steps for the training and testing process. Therefore, the proposed RL agent and the baseline agent train over the 160 simulation rounds and are tested in the remaining 40.

Fig. 2 shows the Cumulative Distribution Function (CDF) of the average served throughput rate for each slice using the proposed method *ssr_protect* and the baseline *ssr* over the test set. The throughput requirements are 20 Mbps and 5 Mbps for eMBB and URLLC slices, respectively. Note that we do not add mMTC throughput requirements since it does not have stringent throughput requirements. The baseline and the proposed method did not fulfill the 20 Mbps requirement for the eMBB slice in most parts of the simulation. Still, the baseline method obtained a higher throughput rate to the eMBB slice than our proposed method. When considering the throughput requirement for URLLC slice, the proposed method fulfilled the 5 Mbps requirement over the entire simulation round, and the baseline accomplished it in almost the entire simulation.

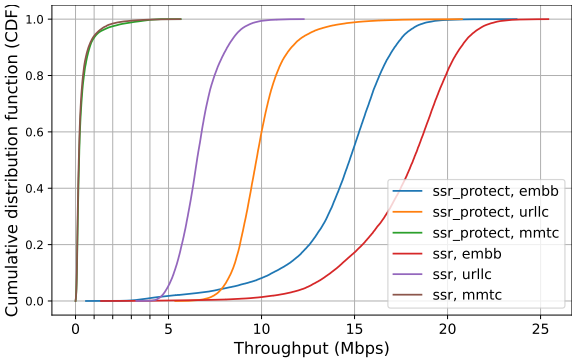

Fig. 2: CDF of the average throughput rate for each slice.

Fig. 3 shows the buffer delay per slice obtained using the baseline and the proposed method. The mMTC and eMBB slice delay requirements were fulfilled by all the RRS agents. Still, when observing the URLLC slices, the proposed method obtained a lower delay value fulfilling the URLLC slice requirement over the entire simulation round, while the baseline method kept the buffer delay above the 1 ms requirement defined to the URLLC slice.

As we can observe, the throughput and delay analyses are insightful but incomplete when investigating the QoS requirements fulfillment. They offer a focused inspection of a specific network requirement while the agent aims to

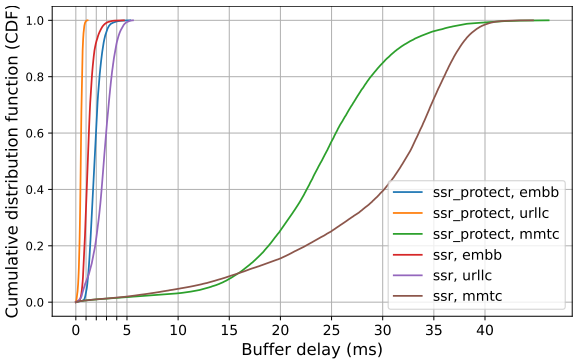

Fig. 3: Buffer delay obtained per slice.

maximize the slice requirements fulfillment. Therefore, an important metric is the number of requirement violations that account for the number of times the network requirement was not fulfilled. Moreover, sometimes the network resources are insufficient to provide the throughput and delay requirements, and the RRS agent should prioritize the most important slices to protect critical applications, represented by the URLLC slice in our simulation.

Fig. 4 shows the average number of violations and the standard deviation using the baseline and the proposed method over the entire test set. The proposed method obtained fewer slice requirement violations than the baseline with greater stability. Furthermore, the number of URLLC violations was expressively reduced in relation to the number of URLLC violations obtained by the baseline method. Therefore, the proposed method was able to prioritize the URLLC slice to protect critical applications and also minimize the total number of violations in the network. Even with the baseline obtaining better performance concerning the served throughput to the eMBB slice, in general, the requirement fulfillment performance of the proposed agent got a better balance of the network metrics by protecting the URLLC slice and obtaining a circumstantial decrease in the number of slice violations.

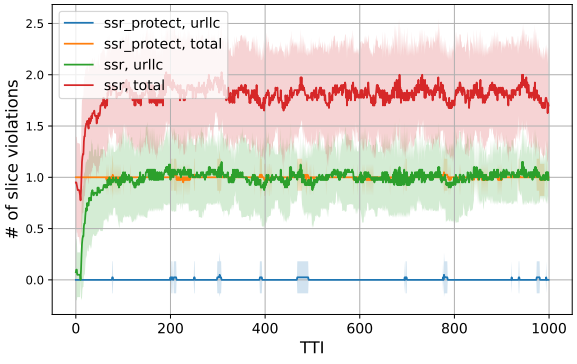

Fig. 4: Average number of violations and the standard deviation.

V. Conclusions

We proposed an RRS RL agent based on the SAC technique for 4.0 Industry considering RAN slicing. The RRS was responsible for orchestrating the radio resources among the slices to fulfill the slice requirements and protect the URLLC slice in relation to the other slices when the number of RBs available is not sufficient to fulfill all slice requirements. The proposed SAC RL agent outperformed the baseline with a smaller number of slice violations and by prioritizing critical applications represented by URLLC slice. This paper showed preliminary results considering only one BS. As future works, we intend to extend this scenario to a multi-cell case as in [14] and deal with inter-cell interference mitigation.

Acknowledgment

This work was supported partially by RNP, with resources from MCTIC, Grant No. 01245.010604/2020-14, under the Brazil 6G Project of Instituto Nacional de Telecomunicações (Inatel), Brazil.

References

[1] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "ColoRAN: Developing machine learning-based xapps for open RAN closed-loop control on programmable experimental platforms," *IEEE Transactions on Mobile Computing*, pp. 1–14, 2022.

[2] Y. Wu *et al.*, "A survey of intelligent network slicing management for industrial IoT: Integrated approaches for smart transportation, smart energy, and smart factory," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1175–1211, 2022.

[3] M. Yan *et al.*, "Intelligent resource scheduling for 5G radio access network slicing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7691–7703, 2019.

[4] J. Mei *et al.*, "Intelligent radio access network slicing for service provisioning in 6G: A hierarchical deep reinforcement learning approach," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6063–6078, 2021.

[5] S. Jaeckel, L. Raschkowski, L. Borner, K. Thiele, F. Burkhardt, and E. Eberlein, "QuaDRiGa - quasi deterministic radio channel generator, user manual and documentations," 2021.

[6] T. Jiang, J. Zhang, P. Tang, L. Tian, Y. Zheng, J. Dou, H. Asplund, L. Raschkowski, R. D'Errico, and T. Jämsä, "3GPP standardized 5g channel model for iiot scenarios: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8799–8815, 2021.

[7] S. Jaeckel *et al.*, "Industrial indoor measurements from 2-6 ghz for the 3GPP-NR and QuaDRiGa channel model," in *IEEE 90th Vehicular Technology Conference*, 2019, pp. 1–7.

[8] X. Li, R. Ni, J. Chen, Y. Lyu, Z. Rong, and R. Du, "End-to-end network slicing in radio access network, transport network and core network domains," *IEEE Access*, vol. 8, pp. 29 525–29 537, 2020.

[9] B. Khodapanah, A. Awada, I. Viering, J. Francis, M. Simsek, and G. P. Fettweis, "Radio resource management in context of network slicing: What is missing in existing mechanisms?" in *Proc. of IEEE Wireless Communications and Networking Conference*, 2019, pp. 1–7.

[10] R. Schmidt, C.-Y. Chang, and N. Nikaein, "Slice scheduling with QoS-guarantee towards 5G," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–7.

[11] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. of International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[12] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[13] R. Li *et al.*, "The LSTM-based advantage actor-critic learning for resource management in network slicing with user mobility," *IEEE Communications Letters*, vol. 24, no. 9, pp. 2005–2009, 2020.

[14] W. V. Fernandes Mauricio, F. R. Marques Lima, A. Taufik, T. Ferreira Maciel, and D. Aguiar Sousa, "Resource allocation for energy efficiency and QoS provisioning," *Journal of Communication and Information Systems*, vol. 34, no. 1, pp. 224–238, Oct. 2019. [Online]. Available: https://jcis.sbrt.org.br/jcis/article/view/665