IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/TMLCN.2022.1234567

Deep Reinforcement Learning for Uplink Scheduling in NOMA-URLLC Networks

Benoît-Marie Robaglia[1], Member, IEEE, Marceau Coupechoux[1], and Dimitrios Tsilimantos[2]

[1]LTCI, Telecom Paris, Institut Polytechnique de Paris
[2]Advanced Wireless Technology Lab, Paris Research Center, Huawei Technologies Co. Ltd.

Corresponding author: Marceau Coupechoux (email: marceau.coupechoux@telecom-paris.fr).

ABSTRACT

This article addresses the problem of Ultra Reliable Low Latency Communications (URLLC) in wireless networks, a framework with particularly stringent constraints imposed by many Internet of Things (IoT) applications from diverse sectors. We propose a novel Deep Reinforcement Learning (DRL) scheduling algorithm, named NOMA-PPO, to solve the Non-Orthogonal Multiple Access (NOMA) uplink URLLC scheduling problem involving strict deadlines. The challenge of addressing uplink URLLC requirements in NOMA systems is related to the combinatorial complexity of the action space due to the possibility to schedule multiple devices, and to the partial observability constraint that we impose to our algorithm in order to meet the IoT communication constraints and be scalable. Our approach involves 1) formulating the NOMA-URLLC problem as a Partially Observable Markov Decision Process (POMDP) and the introduction of an *agent state*, serving as a sufficient statistic of past observations and actions, enabling a transformation of the POMDP into a Markov Decision Process (MDP); 2) adapting the Proximal Policy Optimization (PPO) algorithm to handle the combinatorial action space; 3) incorporating prior knowledge into the learning agent with the introduction of a Bayesian policy. Numerical results reveal that not only does our approach outperform traditional multiple access protocols and DRL benchmarks on 3GPP scenarios, but also proves to be robust under various channel and traffic configurations, efficiently exploiting inherent time correlations.

INDEX TERMS Deep Reinforcement Learning, Internet of Things, Multiple Access, POMDP, Proximal Policy Optimization, URLLC.

I. INTRODUCTION

NEW high-demanding use cases pertaining to various industry sectors need to be addressed by the future generations of wireless networks[1]. In particular, the Third Generation Partnership Project (3GPP) standard [1] has defined Ultra Reliable Low Latency Communications (URLLC) requirements for many Internet of Things (IoT) use cases such as smart grids, factory automation and intelligent transportation to only name a few. A classical URLLC reliability requirement is for example to transmit a 32-byte packet with success probability $1 - 10^{-5}$ and with a latency deadline of 1 ms [1]. A deadline is said to be *strict* if the packet is lost beyond this delay. URLLC requirements are particularly challenging on the uplink, i.e., from IoT devices to a central Base Station (BS), because the BS can acquire traffic and channel information only at the cost of a significant signalling load and delay; a problem related to *partial observability* in control theory. In order to reduce latency and improve reliability, Non-Orthogonal Multiple Access (NOMA) is seen as a promising transmission technique, as it allows to schedule multiple users on the same time-frequency resource and to improve the spectral efficiency [2]. However, even with NOMA, there is in practice a limited number of users sharing the same resource and the *user selection* issue adds to the complexity of the traditional many-to-one scheduling problem. In this context, we thus propose NOMA-PPO, a new Deep Reinforcement Learning (DRL)

[1]The work of B.-M. Robaglia and M. Coupechoux has been performed at the LINCS laboratory (lincs.fr).

scheduling algorithm to solve the NOMA uplink URLLC scheduling problem with strict deadlines.

A. Related Work

1) Uplink URLLC access solutions

Uplink access schemes for URLLC can be divided in two main groups, namely grant-based and grant-free protocols. Both approaches can be extended using NOMA or Deep Reinforcement Learning (DRL).

In the first set, the scheduling of the devices is performed by the BS, see e.g. [3], [4]. Devices with a packet to transmit first send a scheduling request on the uplink. The BS then allocates uplink resources for the packet transmission. Uplink packets may include in their header some scheduling information (like the buffer status) to avoid the scheduling request step. In this case, a scheduling algorithm is required at the BS to meet the delay and reliability constraints without losing resources when a polled device has no packet to transmit. This is the baseline protocol adopted in 5G New Radio (NR) [5]. The main drawback of the approach lies in the duration of the four-way handshake that may be incompatible with URLLC constraints. The advantage, though, is to avoid collisions between device transmissions.

In the second set of access schemes, the handshaking is removed by allowing uplink transmissions to be grant-free (GF). This means that devices can transmit without an explicit command from the BS. We can further distinguish contention-free and contention-based GF access. In contention-free GF (also called semi-persistent scheduling), the BS pre-allocates periodic orthogonal uplink resources to the devices, so that there are no collisions [6]. When a device has a packet to send, it waits for the next opportunity. This access scheme has been also adopted by 5G NR [5]. Contention-free GF is however mostly adapted to periodic deterministic traffic, but becomes inefficient when the traffic is sporadic or probabilistic because resources may be lost, if there is no packet to be sent, or deadlines violated, when the packet arrival rate is suddenly higher.

Several papers have studied contention-based GF, a family of protocols that are versions of Slotted Aloha (SA) enriched with smart retransmission schemes. Contrary to other approaches, uplink transmissions are indeed here subject to collisions. A typical example of this literature is the work presented in [7], where authors adapt SA to URLLC and industrial IoT use cases by introducing retransmission schemes that depend on the traffic profile of the devices. In [8], authors summarize the classical retransmission schemes: the K-repetition GF scheme, in which a pre-determined number of copies of the same packet are transmitted; the reactive GF scheme, in which devices receive a feedback from the BS for every transmission; and the proactive GF scheme, in which a packet is repeatedly sent until a positive acknowledgement is received. These protocols have been enhanced using NOMA [2] with the goal of better using the available resources and reduce the number of collisions for a given traffic load, see e.g. [8]–[10] and references therein. However, with or without NOMA, all SA-based approaches suffer from high collision rates when the load or the number of devices increases [11] and fail to take advantage of the various traffic patterns or channel conditions across the devices.

Recent advances in Deep Reinforcement Learning (DRL) [12] have been applied to solve several limitations in IoT systems [13] and are potential solutions for the aforementioned problems. Several proposals use Deep Multi-Agent Reinforcement Learning (MARL) to model a user with a DRL algorithm in order to learn a transmission protocol in a decentralized manner in the context of dynamic spectrum access [14]–[16]. Nonetheless, these solutions do not tackle the URLLC constraint with strict deadlines and do not take into account the potential of NOMA. The approach of [17] models the massive access problem by transforming the URLLC constraint into a data rate constraint and learns a transmission strategy in order to maximize the network energy efficiency using cooperative MARL. However, the authors do not consider strict deadlines and do not address the theoretical limitations of decentralized MARL like the non-stationarity during training.

At last, several strategies leveraging DRL have been put out to deal with the URLLC constraint in NOMA systems. The authors of [18] propose Deep-SARSA to tackle the resource allocation problem at the BS for minimizing the error probability in uplink transmissions. Yet, the proposed solution does not take into account the packet arrival processes, assumes full observability of the system and does not impose strict deadlines. Additionally, the work of [19] optimizes a NOMA based GF protocol with DRL. The authors use DRL to dynamically adjust the number of repetitions and radio resources in the proactive GF scheme. Nevertheless, the approach, which is based on SA, still suffers from a high collision rate as the load increases, is not designed for handling both deterministic and sporadic traffic and fails to take advantage of channel correlations. A very preliminary version of our work has been presented in [20]. However, the proposed solution ignores NOMA, does not take into account channel correlations and is only adapted to probabilistic periodic traffic.

In this paper, we consider a system in which the BS semi-blindly schedules the devices for their uplink transmissions, as it is done in grant-based access, however without the need for scheduling requests. Thanks to NOMA, the BS is able to poll multiple devices for a transmission in the same resource. We thus tackle a partially observable scheduling problem where the BS should strike a balance between acquiring scheduling information and avoiding excessive collisions. Our problem is characterized by two challenges, namely a combinatorial action space and a partially observable environment, that conventional DRL algorithms fail to handle.

2) DRL challenges for uplink URLLC

First, allowing the BS to poll multiple devices in a frame drastically increases the action space. For k devices, the decision maker needs to choose between 2^k actions, which is exponential in k. Few solutions have been proposed to address this problem in the literature. The most common one is proposed in [21]. The authors' idea is to project the large discrete action space in a continuous action space and thus solve a continuous action RL problem with the traditional Deep Deterministic Policy Gradient algorithm [22]. However, this approach assumes that the discrete action space can be embedded in a continuous space, which is not straightforward for our Multiple Access (MA) problem. An alternative is the work of [23]. The authors solve a high dimensional action space RL problem with a Recurrent Neural Network (RNN) to sequentially predict the action vector, one dimension after the other. Nevertheless, not only does this algorithm assume that we know how to order the action dimensions, but the Q-value estimated for the last dimension is very noisy, especially when k is large. An extension of this paper is the Branching Dueling Q-Network (BDQ) [24]. The authors solve a RL problem with a k-dimensional action space using a dueling architecture where there is a value network common for all dimensions and k advantage networks, one for every dimension. Yet, not only is this solution ill-suited to manage partial observability, but it also cannot account for any prior knowledge the agent might have regarding the dynamics of the environment.

Second, as the BS is not aware of the whole environment and takes decisions solely based on partial observations of the environmental state, our problem can be modeled by a POMDP [25]. When observations are not Markovian, traditional RL algorithms work with history dependent policies, an approach that can be rapidly computationally intractable as the number of possible histories grows exponentially with the horizon. A way to alleviate this problem is to introduce *belief states*, a probability distribution over the states, which is also a sufficient statistic for the past history and the initial state distribution. A POMDP can be then reformulated as a MDP in which the state space is the continuous belief state space. Traditional RL methods like Q-learning or policy gradient algorithms can finally be used on the resulting belief-MDP [26].

Three main methods are proposed in the literature to derive or estimate a belief state: 1) the belief update formula [26], 2) a RNN [27] and 3) a generative model [28]. However, all these methods suffer from major drawbacks. While the belief update formula requires the knowledge of the environment dynamics (transition and observation function), using a RNN or a generative model introduces a new layer of complexity since there are now two phases involved: the belief estimation and the computation of the optimal policy. Additionally, since Deep Neural Networks (DNNs) are black boxes, it is impossible to add any prior knowledge that the agent might have about the environment. Moreover, the learned beliefs are difficult to interpret and error might be propagated to the policy optimization phase.

An alternative to the belief state is the notion of *information state* [29] or *internal state* [30, Section 12.4.2]. The idea is to derive a function of the history which is a sufficient statistic for estimating the environmental state. However, learning such a sufficient representation of the history is difficult as it is often task-specific.

B. Contributions and outline

In this paper, we formulate the NOMA-URLLC problem as a partially-observable scheduling problem and solve it by proposing a DRL algorithm. Our contributions can be summarized as follows:

- We formulate a general MA problem with the URLLC constraint, considering packets with strict deadlines and NOMA uplink communications as a POMDP.
- We introduce the notion of *agent state* in order to theoretically address the POMDP formulation. We show that the agent state is a sufficient statistic for the past observation-action history that allows us to 1) express past actions and observations in a compact way, and 2) convert the POMDP problem to an MDP and benefit from the convergence properties of the DRL algorithms. This transformation can be extended to other wireless settings where partial observability regarding the buffer or channel evolution needs to be addressed.
- We propose a DRL algorithm, *NOMA-PPO*, that enhances the state-of-the-art algorithm PPO [31] with two components: 1) a branching policy network architecture in order to linearly manage combinatorial action spaces. This idea is inspired by the BDQ architecture [24] and extended to PG methods. 2) Bayesian policies, that incorporate prior information about the MA problem into the DRL agent [32].
- We provide numerical evidence that our approach outperforms traditional MA and DRL benchmarks across 3GPP scenarios in terms of URLLC score, convergence speed, and fairness. Furthermore, we show that our algorithm is able to cope with different traffic models, a deterministic periodic and a probabilistic aperiodic traffic model in particular. Finally, our algorithm exhibits robustness against different channel configurations and demonstrates a successful exploitation of time-varying channel information.[2]

In Section II, we define the system model. Section III formulates the POMDP problem. Section IV presents the NOMA-PPO approach and finally, Section V exposes the simulations and numerical results.

Notations: For a finite set X, $\Delta(X)$ denotes the set of all probability distributions over X. The indicator function is denoted $\mathbb{1}\{\cdot\}$, $diag(\cdot)$ is the diagonal operator that transforms a

[2]The code associated with the research presented in this paper is available for public access on the following GitHub repository.

vector in a diagonal matrix and $\odot$ is the Hadamard product. The matrices are written in bold upper case and the vectors in bold lower case. $\langle\cdot\rangle$ refers to a tuple, $[\cdot]$ to the modulo operator and $\arg_B \min(S)$ returns a set of B elements in S having the lowest value (ties are broken at random). The system model parameters are summarized in Table 1, while those for the algorithm can be found in Table 2.

II. System Model

A. Network Model

We consider a time-slotted wireless network of K heterogeneous devices communicating with a BS over a wireless shared channel on the uplink. Every device has a single antenna and the BS is equipped with n_a antennas. The time is divided into radio frames of duration T_f and every frame is divided into five time-slots of duration T_s (see Fig. 1). This division represents the minimum time required for the processes of polling, transmitting and acknowledging. The time synchronization among all devices is performed by the BS using downlink signals. During the first slot of every radio frame, the BS is allowed to poll a number of devices for a potential uplink transmission, described by the vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_K) \in \{0,1\}^K$, where $a_k = 1$ when the device k is polled and $a_k = 0$ otherwise. It also allocates orthogonal resources for uplink pilot transmissions from the polled devices. After a guard interval, a *polled* device with at least a packet in its buffer becomes *active* and transmits during the third slot. Its transmission includes a pilot signal for channel estimation, sent using the orthogonal resource allocated by the BS. Its transmission also includes the buffer status of the device. Polled devices without any packet to transmit do not send any pilot and thus leave the allocated resource unused. We assume that all packets have the same size of L bits. After a guard interval, the BS acknowledges the reception of successful transmissions.

The set of active users at frame $t \in \mathbb{N}$ is denoted $\mathcal{U}(t)$ and the number of active devices is denoted $U(t)$, i.e., $|\mathcal{U}(t)| = U(t)$. We denote also $\boldsymbol{u}(t) \in \{0,1\}^K$ the vector of active users at frame t such that $u_k(t) = \mathbb{1}\{k \in \mathcal{U}(t)\}$. Besides, we define $\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$, $\boldsymbol{\tau}^s(t)$ vectors of size K, where each component k represents the number of frames since the last time device k has been polled, active and successfully decoded, respectively.

We assume that the system is using NOMA [2] to improve the spectral efficiency of the network. NOMA allows several users to use the same frequency and time resources by superposing their signal in the power domain. At the receiver side, the BS applies Successive Interference Cancellation (SIC) to decode the superposed signals.

B. Interference Channel Model

We adopt a realistic channel model that has been adopted in the literature, based on the evaluation of the Signal-to-interference-plus-noise ratio (SINR) [33] and the finite block length regime, see e.g. [34].

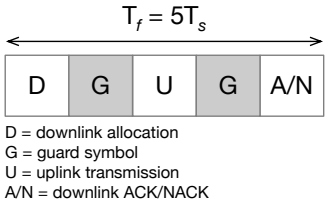


FIGURE 1: Radio frame structure.

1) Received Signal

In this model, a device $k \in \mathcal{U}(t)$, active in frame t, transmits a signal $s_k(t)$ of power $p_k(t) = \mathbb{E}[||s_k(t)||^2]$, where the expectation is taken over possible symbols. In a general formulation, transmit power could be controlled. However, for the sake of simplicity, this work focuses on scenarios with a fixed transmit power. The BS is supposed to receive the signal with n_a antennas and to perform Maximum Ratio Combining (MRC). The transmission of user k experiences a large scale fading $g_k(t)$, which accounts for the distance-dependent path-loss and shadowing, fast fading $\boldsymbol{h}_k(t) = [h_{k1}(t), \cdots, h_{kn_a}(t)]^T \in \mathbb{C}^{n_a \times 1}$ and thermal noise $\boldsymbol{n} \in \mathbb{C}^{n_a \times 1}$. The signal received by the BS at frame t from all active devices can thus be written as a superposition of $s_1(t), \ldots, s_{U(t)}(t)$ and thermal noise:

$$\boldsymbol{r_s}(t) = \sum_{k \in \mathcal{U}(t)} \boldsymbol{h}_k(t)\sqrt{g_k(t)}s_k(t) + \boldsymbol{n}(t) \tag{1}$$

where $n_i(t), i = 1, \ldots, n_a$ is an independent circularly symmetric white Gaussian process with distribution $\mathcal{CN}(0, \sigma_n^2 \boldsymbol{I})$. Thanks to the orthogonal pilots sent on the uplink, the BS is able to estimate the channel realizations of active devices. From now on, we assume that the BS has a perfect channel state information for decoding. In MRC, the signals received on the n_a antennas are combined using a weight vector $\boldsymbol{w}_k^H = \boldsymbol{h}_k$ for device k. The combined signal $\boldsymbol{y}_k(t) = \boldsymbol{w}_k^H \boldsymbol{r_s}$ for device k is thus:

$$\begin{aligned}\boldsymbol{y}_k(t) = \boldsymbol{h}_k^H(t)\boldsymbol{h}_k(t)\sqrt{g_k(t)}s_k(t) + \boldsymbol{h}_k^H(t)\boldsymbol{n}(t) \\ + \sum_{j \in \mathcal{U}(t)\setminus\{k\}} \boldsymbol{h}_k^H(t)\boldsymbol{h}_j(t)\sqrt{g_j(t)}s_j(t)\end{aligned} \tag{2}$$

We assume that BS antennas are sufficiently spaced so that the fading coefficients at every antenna are spatially uncorrelated and thus $\boldsymbol{h}_k \sim \mathcal{CN}(0, \boldsymbol{I})$ for all k. The fast fading process $h_{ki}(t)$, for $k = 1, ..., K$ and $i = 1, ..., n_a$, is supposed to follow a time-correlated Gauss-Markov model [35]:

$$h_{ki}(t) = \bar{a}_k h_{ki}(t-1) + z_k(t) \tag{3}$$

where $z_k(t) \sim \mathcal{CN}(0, 1 - \bar{a}_k^2)$. The fading correlation coefficient $\bar{a}_k$ is modeled using the Jakes' model [36]: $\bar{a}_k = J_0(2\pi v_k f_c T_f / c)$, where J_0 is the Bessel function of the first kind and order 0, v_k is the speed of device k, f_c is the carrier frequency, c is the speed of light and $h_{ki}(0) \sim \mathcal{CN}(0,1)$. The coherence time for a device moving at speed v is $T_c = c/(8 f_c v)$ [37]. The channels are supposed to be mutually independent across devices and constant during a frame

TABLE 1: System Model Parameters

Notation	Description
K, n_a	Number of devices, number of antennas.
T_s, T_f	Slot length, radio frame length.
$\mathcal{U}(t)$, $\boldsymbol{u}(t)$	Set and vector of active devices at t.
$s_k(t)$, $p_k(t)$	Transmitted signal, power of k at t
$g_k(t)$, $\boldsymbol{h}_k(t)$	Large scale fading, fast fading of k at t
$\boldsymbol{r}_s(t)$	Received signal from active users.
$\boldsymbol{y}_k(t)$	Combined signal for k at t.
$\alpha(t)$	Decoding order permutation.
$\phi_k(t)$	Indicator for successful decoding of k.
$\eta_k(t)$	Received power for k at t.
$\gamma_k^{\text{no-SIC}}(t)$, $\gamma_k^{\text{SIC}}(t)$	SINR for k without SIC, with SIC at t.
$\epsilon_k(t)$	Error probability of k at t.
n, $M^*(n,\epsilon)$	block length, maximum code size.
B	SIC limitation.
f_k, N_p	Offset, period of packet generation for k.
d_k^h, δ_k	Head-of-line delay, latency constraint of k.
$\bar{\xi}_k(t\|f_k,\xi_k,N_p)$	Probability that k generates a new packet at t.
λ_k	Rate of packet generation at k.
$\boldsymbol{B}(t)$, $\boldsymbol{b}_k(t)$	Buffer status matrix and vector of k at t.
$\boldsymbol{d^h}(t)$	Head-of-line delays of users at t.
$\mathcal{T}^B$, $\mathcal{T}^H$	Buffer and channel state transition operators.

TABLE 2: Algorithm Parameters

Notation	Description
$\boldsymbol{s}(t)$, $\boldsymbol{o}(t)$	State and observation at t.
$\boldsymbol{B}^o(t)$	Observed buffers at t.
$\boldsymbol{\eta}^o(t)$	Observed received power at t.
$\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$, $\boldsymbol{\tau}^s(t)$	Last time the devices have been polled, active, successfully decoded.
$\boldsymbol{a}(t)$	Action of the agent at t.
$r(t-1)$	Reward at step $t-1$.
f^A	Transition function for the agent state.
$\boldsymbol{B}^A(t)$	Buffers representations by the agent.
$\boldsymbol{\eta}^A(t)$	Last known received power by the agent.
$\hbar(t)$	History of observations and actions at t
π_θ	Policy parameterized by θ.
$A^{\pi_{\theta_{\text{old}}}}$,$V^{\pi_{\theta_{\text{old}}}}$	Advantage and value functions.
V_φ	Value network parameterized by φ
$\hat{A}^{GAE}(t)$	Generalized Advantage Estimator.
ν	Clipping parameter for PPO.
γ	Discount factor.
EDF	EDF prior.
f_{ch}	Channel prior.
f	Bayesian prior over the agent state.
q	Posterior policy

(following a block fading channel model [38]). We assume a rich scattering environment with stationary scatterers, as detailed in [35], [36], [38]. We denote $\boldsymbol{H}(t) \in \mathbb{C}^{n_a \times K}$ the matrix of all channel realizations at time t and $\mathcal{T}^H$ the evolution process, i.e., $\boldsymbol{H}(t+1) \sim \mathcal{T}^H(\boldsymbol{H}(t))$.

2) Decoding Order

The SIC decoding order at each frame t can be seen as a permutation function $\alpha(t)$ over the set of active devices, i.e., $\alpha(t) : [1 : U(t)] \to \mathcal{U}(t)$. For any $i = 1, ..., U(t)$, $\alpha_i(t)$ is the i-th decoded device's index and for any $k \in \mathcal{U}(t)$, $\alpha_k^{-1}(t)$ is the rank of user k in the decoding process. When the BS tries to decode device $\alpha_i(t)$, it has already tried to decode all devices $\alpha_1(t)$,..., $\alpha_{i-1}(t)$. Each decoding might have been successful or not. Let $\phi_k(t)$ be the indicator whether an active device $k \in \mathcal{U}(t)$ has been successfully decoded by the BS ($\phi_k(t) = 1$) or not ($\phi_k(t) = 0$) and $\boldsymbol{\phi}(t) = (\phi_1(t), \ldots, \phi_K(t))$ the vector of all indicators. As a consequence, the signal received at the BS from $\alpha_i(t)$ is subject to the interference of $\alpha_j(t)$, $j > i$, i.e., from devices that have not been yet considered for decoding by the BS, and to the interference of $\alpha_j(t)$, $j < i$ whenever $\phi_{\alpha_j(t)} = 0$, i.e., from devices that have not been successfully decoded by the BS.

We now assume the decoding order that minimizes the total transmit power, given target rates on the uplink [37]: active devices are sorted in decreasing order of their received power at the BS, as follows:

$$\eta_{\alpha_1}(t) \geq \eta_{\alpha_2}(t) \geq \cdots \geq \eta_{\alpha_{U(t)}}(t) \tag{4}$$

where $\eta_k(t) = p_k(t) g_k(t) ||\boldsymbol{h_k}(t)||^2$. We denote $\boldsymbol{\eta}(t) = (\eta_1(t), \eta_2(t), \ldots, \eta_K(t))$ the vector of received powers. We denote $\boldsymbol{\eta}^o(t)$ the vector of powers received by the BS from the active devices, observed at time t thanks to the transmitted pilots, i.e., $\boldsymbol{\eta^o}(t) = \text{diag}(\boldsymbol{u}(t))\boldsymbol{\eta}(t)$.

3) Signal to Interference plus Noise Ratio

In absence of SIC, the signal (2) results in a SINR at the output of the combiner [39]:

$$\gamma_k^{\text{no-SIC}}(t) = \frac{\eta_k(t)}{\sum_{j \neq k} \eta_{jk}(t) + \sigma_n^2} \tag{5}$$

where $\eta_{jk}(t) = p_j(t) g_j(t) \frac{|\boldsymbol{h}_k^H \boldsymbol{h}_j|^2}{||\boldsymbol{h}_k||^2}$.

With SIC however, we decode in the decreasing order of $\eta_k(t)$, so that part of the interference is potentially successively removed. The SINR with SIC writes now:

$$\gamma_k(t) = \frac{\eta_k(t)}{\underbrace{\sum_{j \in J_1} (1 - \phi_j(t)) \eta_{jk}(t)}_{\text{before } k \text{ in decoding order}} + \underbrace{\sum_{j \in J_2} \eta_{jk}(t)}_{\text{after } k} + \sigma_n^2} \tag{6}$$

where $J_1 = \{j \in \mathcal{U}(t), \alpha_j^{-1}(t) < \alpha_k^{-1}(t)\}$ is the set of devices that are considered for decoding before k and $J_2 = \{j \in \mathcal{U}(t), \alpha_j^{-1}(t) > \alpha_k^{-1}(t)\}$ is the set of devices that are decoded after k. Note that ϕ_j is determined iteratively: we

are able to compute the SINR of device k, only once we know the outcome of the decoding for devices $j \in J_1$.

4) Achievable Rate with Finite Block Length

As the URLLC messages are often supposed to be very small [40] (in the factory automation scenario for instance), we adopt a finite block-length regime [41] for the calculation of the achievable rate. In this model, an encoder maps every L-bit message $m \in [1 : M]$ to a codeword $c_m \in \mathbb{C}^n$, where $M = 2^L$ is the size of the message space and n is the block length, also known as the number of complex channel uses. Codewords are subject to an average power constraint, i.e., $\frac{1}{M}\sum_m ||c_m||^2 = n\eta$, where η is the received power per channel use. The codeword is transmitted over an Average White Gaussian Noise channel with noise variance σ^2. At the receiver, a decoder maps the channel output to an estimate $\tilde{m}$ of the message. The average error probability is defined as $\varepsilon = \mathbb{P}[\tilde{m} \neq m]$. A codebook $\{c_m \in \mathbb{C}^n, m \in [1 : M]\}$ and a decoder whose average error probability is less than ε are called a (M, n, ε)-code. For given ε and n, the maximum code size is denoted $M^*(n, \varepsilon)$. Authors of [41] provide a normal approximation of the maximum achievable code rate:

$$\frac{\log_2 M^*(n,\varepsilon)}{n} \approx C(\gamma) - \sqrt{\frac{V(\gamma)}{n}} Q^{-1}(\varepsilon) \quad (7)$$

where $\gamma = \frac{P}{\sigma^2}$ is the Signal to Noise Ratio (SNR), P is the received power, $C(\gamma) = \log_2(1+\gamma)$ is the Shannon capacity[3], $V(\gamma) = \frac{\gamma}{2}\frac{\gamma+2}{(\gamma+1)^2}\log_2^2 e$ is the channel dispersion and $Q(x) = 1/\sqrt{2\pi}\int_x^\infty \exp(-t^2/2)dt$. Although (7) is an asymptotic approximation when n tends to infinity, it is tight for n as small as 200 [41]. When considering a block fading channel with channel realization h, (7) is valid *conditionnally* to the channel realization with $\gamma = \frac{P|h|^2}{\sigma^2}$. In our study, we further treat interference as noise, as it is usually done in the literature [38, Chapter 15], and apply (7) with the SINR in (6). As a consequence, a packet of device k, transmitted at frame t, is not successfully decoded with probability:

$$\epsilon_k(t) = Q\left(\sqrt{\frac{n}{V(\gamma_k(t))}}\left(C(\gamma_k(t)) - \frac{L}{n}\right)\right) \quad (8)$$

The downlink allocation and the acknowledgment are supposed to be error free.

5) SIC Limitation

We define an upper limit B on the number of possible multiplexed users which is characteristic of the SIC performance [37]. More specifically, a necessary condition for a device k to be decoded is that the number of active devices is less than B, i.e.

$$|\mathcal{U}(t)| \leq B \quad (9)$$

[3]Note that contrary to [41], which considers the capacity per real dimension, we use a complex representation of the signal. As a consequence, C is the capacity per complex dimension [37, Chapter 5].

In other words, any simultaneous transmission of more than B packets provokes a collision.

Finally, at frame t and for a user k, we can write $\phi_k(t)$ as a Bernoulli random variable of parameter $1 - \epsilon_k(t)$, i.e., $\phi_k(t) \sim \mathcal{B}(1 - \epsilon_k(t))$ when (9) is satisfied, and $\phi_k(t) = 0$ otherwise. The SIC decoding procedure is summarized in Algorithm 1.

Algorithm 1: SIC Decoding Procedure

input : $\mathcal{U}(t)$, $\boldsymbol{\eta}(t)$
output: $\boldsymbol{\phi}(t) = (\phi_1(t), \phi_2(t), \ldots, \phi_K(t))$
Initialize: $\phi_k(t) = 0, \forall k$.
if $|\mathcal{U}(t)| \leq B$ **then**
  foreach $k \in \mathcal{U}(t)$ *in decreasing order of* $\boldsymbol{\eta}(t)$ **do**
    Compute the SINR $\gamma_k(t)$ using (6) and $\boldsymbol{\phi}(t)$
    Compute $\epsilon_k(t)$ using $\gamma_k(t)$ according to (8)
    Draw $\phi_k(t)$ from the Bernoulli distribution: $\phi_k(t) \sim \mathcal{B}(1 - \epsilon_k(t))$, i.e. decode the packet with probability $1 - \epsilon_k(t)$

C. Traffic Models

Packets are generated at the devices according to models of either probabilistic periodic traffic or probabilistic aperiodic traffic and are subject to a strict deadline constraint.

Probabilistic periodic traffic

In this model, directly inspired by [42], a device k generates packets periodically every N_p radio frames with probability ξ_k. Devices are not synchronous, i.e., each device is assigned an offset parameter $f_k \in [0, N_p]$ such that, at every radio frame $t \geq 0$, the probability for a device k of generating a new packet is: $\bar{\xi}_k(t|f_k, \xi_k, N_p) = \mathbb{1}_{\{t[N_p]=f_k\}}\xi_k$. Note that a specific case for this model is the *deterministic periodic traffic* as defined in [40, Annex A] for various use cases including for example factory automation, where $\xi_k = 1$ and $f_k = 0$ for all k. Periodic transmissions may correspond to the periodic update of a position or the repeated monitoring of a characteristic parameter [43].

Probabilistic aperiodic traffic

This traffic model is defined in [40] and is based on the File Transfer Protocol (FTP) model 3 defined in [44], however with a fixed packet length. At every device k, packets are generated according to a Poisson process of rate λ_k. An aperiodic transmission may correspond to process, diagnostic or maintenance events that trigger the transmission [43].

Deadlines

Every device k has an individual latency constraint $\delta_k \in \mathbb{N}^*$ expressed in number of radio frames, such that a packet that has not been transmitted after δ_k radio frames is dropped. Let

$\delta = \max_k \delta_k$. When a transmission fails, a device is allowed to retransmit the packet as long as it has not expired.

Buffers

We assume that devices have an infinite buffer and packets in the queue are delivered in a "first come, first served" manner. For every device k, the buffer at time t can be represented by a vector $\boldsymbol{b}_k(t) = [b_{k,1}(t), ..., b_{k,\delta}(t)] \in \mathbb{N}^\delta$ where $b_{k,d}(t) = i$ when device k has i packets with time-to-deadline d at time t. We denote $\boldsymbol{B}(t)$ the matrix of all buffer status at time t. The *head-of-line delay* $d_k^h(t)$ of user k at frame t is defined as $b_{k,d_k^h(t)}(t) \neq 0$ and $b_{k,d}(t) = 0$ for all $d < d_k^h(t)$ if it has at least one packet in its buffer. This is the smallest time-to-deadline in the buffer of device k. We note $\boldsymbol{d^h} = [d_1^h, d_2^h, \dots, d_K^h]$ the vector of all head-of-line delays. When a device is polled and has at least one packet to transmit, it chooses for transmission one of the packets associated to its head-of-line delay at random.

For a device k, the buffer status transits as follows: (a) Successfully decoded packets are removed from the buffer, i.e. $b_{k,d_k^h(t)-1}(t+1) = b_{k,d_k^h(t)}(t) - 1$ if $\phi_k(t) = 1$; (b) Other packets see their time-to-deadline decreased by one, i.e., $b_{k,d-1}(t+1) = b_{k,d}(t)$ for all $d > 1$. If $d = 1$, the packets expire and are removed from the buffers; (c) If m new packets are generated at the device, they enter the buffer with a deadline δ_k, i.e., $b_{k,\delta_k}(t+1) = m$.

We denote this operation $\mathcal{T}^B$, i.e.,

$$\boldsymbol{B}(t+1) \sim \mathcal{T}^B(\boldsymbol{B}(t), \boldsymbol{\phi}(t)) \tag{10}$$

When an active device is successfully decoded, its buffer status is known (or observed) to the BS. When a device is polled and yet has no packet to transmit, it does not transmit its pilot so that the BS is informed that its buffer is empty. We denote $\boldsymbol{B}^o(t)$ the matrix of observed buffer status at time t.

III. Problem Formulation

A. Optimization Problem

The objective of the BS is to maximize the expected number of successful transmissions with respect to the stochastic policy π that maps the current observation history at frame t: $(\boldsymbol{B}^o(t), \boldsymbol{\eta}^o(t), \dots, \boldsymbol{B}^o(0), \boldsymbol{\eta}^o(0))$ to the vector of devices to schedule $\boldsymbol{a}(t)$. Moreover, buffers and channels are subject to the dynamics $\mathcal{T}^B$ and $\mathcal{T}^H$, respectively. The optimization problem (P) can thus be formulated as:

$$\begin{aligned} \max_{\pi} \quad & \mathop{\mathbb{E}}_{(\mathcal{T}^B, \mathcal{T}^H, \pi)} \left[\sum_{t=0}^{\infty} \sum_{k \in \mathcal{U}(t)} \gamma^t \phi_k(t) \right] \\ \text{s.t. } & \boldsymbol{B}(t+1) \sim \mathcal{T}^B(\boldsymbol{B}(t), \boldsymbol{\phi}(t)) \\ & \boldsymbol{H}(t+1) \sim \mathcal{T}^H(\boldsymbol{H}(t)) \end{aligned} \tag{P}$$

where $\gamma \in [0, 1)$ is the discount factor that determines the importance of future rewards compared to immediate ones.

B. POMDP Formulation

To solve our problem, we adopt the POMDP framework, see e.g. [25], [26]: at every time step t, an agent interacts with the environment by taking an action, gets an observation from the environment and obtains a reward.

Definition 1 (POMDP):
A POMDP can be described by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O})$, where

- $\mathcal{S}$ is the state space, i.e., a finite set of environmental states,
- $\mathcal{A}$ is the action space, i.e., a finite set of actions,
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$ is the transition function which is a probability distribution over the next environmental state s' when it was in state s and the action a has been taken. It verifies the Markov property: $s' \sim \mathcal{T}(\cdot | s(t) = s, a(t) = a)$,
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, where $\mathcal{R}(s, a)$ is the immediate reward by taking action a in state s. We denote $r(t)$ the immediate reward at time t.
- Ω is the observation space, i.e., a finite set of observations,
- $\mathcal{O} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\Omega)$ is the probability distribution of the observation o when the environment is in state s' and the agent has taken action a: $o \sim \mathcal{O}(\cdot | s(t+1) = s', a(t) = a)$.

The *history* $\hbar(t)$ at time t is defined as the sequence of actions taken by the agent and observations from the environment $\hbar(t) = \{o(0), a(0), o(1), a(1), ..., a(t-1), o(t)\}$, where $a(t) \in \mathcal{A}$ and $o(t) \in \Omega$ for all t. The agent makes decisions using a stochastic *policy* π that is a distribution over the actions knowing the history.

The POMDP related to our problem consists in the following components.

1) State space

At each step t, a state $\boldsymbol{s}(t)$ is defined as the concatenation of the buffer status $\boldsymbol{B}(t)$, the received powers $\boldsymbol{\eta}(t)$, and the observation $\boldsymbol{o}(t)$ obtained from the active users at the previous step:

$$\boldsymbol{s}(t) = \langle \boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t) \rangle \tag{11}$$

where $\boldsymbol{o}(t) = \langle \boldsymbol{u}(t-1), \boldsymbol{\phi}(t-1), \boldsymbol{B}^o(t-1), \boldsymbol{\eta}^o(t-1), r(t-1) \rangle$ is the vector of active users, the vector of decoded packets, the observed buffers, the observed received power and the reward at $t-1$ respectively.

2) Action space

The agent has the possibility to poll any subset of devices at every frame. The action space is thus defined as $\mathcal{A} = \{0, 1\}^K$. For $\boldsymbol{a} = (a_1, a_2, \dots, a_K) \in \mathcal{A}$, $a_k = 1$ if the agent

polls device k and $a_k = 0$ otherwise. Note that the action space grows exponentially with the number of devices.

3) Transition function

When the system is in state $\boldsymbol{s}(t)$ at the beginning of a radio frame t, it transits to state $\boldsymbol{s}(t+1)$ at the end of the radio frame. The received power $\boldsymbol{\eta}(t)$ evolves with the channel realizations $\boldsymbol{H}(t)$ and is governed by $\mathcal{T}^H$. Finally the evolution of the buffers is described by $\mathcal{T}^B$. The next observation $\boldsymbol{o}(t+1)$ is computed using the observation function defined in the next subsection.

4) Observation space and observation function

At every frame t, the RL agent can only observe the last feedback from the active users: the set of active users, their channel realizations, the buffer status of successfully decoded devices and the reward. From the state $\boldsymbol{s}(t+1)$ and action $\boldsymbol{a}(t)$, the observation at time $t+1$ is deterministic and defined by:

$$\boldsymbol{o}(t+1) = \mathcal{O}(\boldsymbol{s}(t+1), \boldsymbol{a}(t)) \tag{12}$$

$$= \langle \boldsymbol{u}(t), \boldsymbol{\phi}(t), \boldsymbol{B}^o(t), \boldsymbol{\eta}^o(t), r(t) \rangle \tag{13}$$

In particular, $\boldsymbol{\phi}(t) \sim \mathcal{B}(1-\boldsymbol{\epsilon}(t))$, $\boldsymbol{B^o}(t) = \text{diag}(\boldsymbol{u}(t) \odot \boldsymbol{\phi}(t))\boldsymbol{B}(t)$ and $\boldsymbol{\eta^o}(t) = \text{diag}(\boldsymbol{u}(t))\boldsymbol{\eta}(t)$. Note that when a device k is active but its packet is not decoded, the agent still has the information that this device has a packet to transmit through $u_k(t) = 1$.

5) Reward function

We define the reward function as the number of successfully decoded packets:

$$\mathcal{R}(\boldsymbol{s}(t), \boldsymbol{a}(t)) = \sum_{k \in \mathcal{U}(t)} \phi_k(t) \tag{14}$$

Note that unlike most RL approaches for multiple access [14], [15], [45], we do not penalize the agent when there is a collision or interference. The reason is that we want the agent to learn a tradeoff between sensing and transmitting. In our experiments, we have noticed that using a penalty for collisions did not improve the performance.

The POMDP formulated above aims at maximizing the optimization problem (P). In general, POMDP problems are known to be PSPACE-complete [46], which means that they can be solved using a polynomial amount of memory space and are at least as hard as every other PSPACE problem. In order to solve this POMDP, we introduce a sufficient statistic for the history of past actions and observations, that we call the *agent state*, and that allows us to transform the POMDP problem into an MDP.

C. Agent state for Solving a POMDP

Definition 2 (Agent state):

At the beginning of each frame $t \geq 1$, we define the *agent state* $\boldsymbol{A}(t)$ after the agent receives its observation $\boldsymbol{o}(t)$ as:

$$\boldsymbol{A}(t) = \langle \boldsymbol{B}^A(t), \boldsymbol{\eta}^A(t), \boldsymbol{\tau}^p(t), \boldsymbol{\tau}^a(t), \boldsymbol{\tau}^s(t), r(t-1) \rangle, \tag{15}$$

where $\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$ and $\boldsymbol{\tau}^s(t)$ are the number of frames from t since the last time the devices have been polled, active and have successfully transmitted respectively. $\boldsymbol{\eta}^A(t)$ is the last known received power of the active devices, i.e., its k-th column is $\boldsymbol{\eta}_k(t - \tau_k^a(t) - 1)$. The matrix $\boldsymbol{B}^A(t)$ is a representation of the buffers given the observations made by the BS at time t and is defined as follows. If an active user k has been successfully decoded in the previous frame, we update $\boldsymbol{b}_k^A(t)$ with the new observation, i.e., $\boldsymbol{b}_k^A(t) = \boldsymbol{b}_k^o(t-1)$. For all devices, we decrease the deadlines of the packets in the buffers representation at time $t-1$ by 1 and we remove the expired packets.

While $\boldsymbol{B}^o(t)$ is an immediate observation of the buffers of the active users at t, $\boldsymbol{B}^A(t)$ is a compact representation of all past buffer observations at t. Introducing $\boldsymbol{B}^A(t)$ allows us to incorporate the knowledge of the dynamics of the buffers in the agent. Yet, the agent is still not aware of the new arrivals. To summarise, the agent state at frame t, $\boldsymbol{A}(t)$, can be written as a function f^A of the observation $\boldsymbol{o}(t)$, the previous action $\boldsymbol{a}(t-1)$ and the previous agent state $\boldsymbol{A}(t-1)$, i.e.,

$$\boldsymbol{A}(t) = f^A(\boldsymbol{A}(t-1), \boldsymbol{o}(t), \boldsymbol{a}(t-1)) \tag{16}$$

Proposition 3.1:
$\boldsymbol{A}$ is a sufficient statistic for the action-observation history, i.e.,

$$P(s(t)|\hbar(t)) = P(s(t)|A(t)) \tag{17}$$

Proof:
See Appendix A. ■

Proposition 3.2:
The tuple $(\mathcal{S}^A, \mathcal{A}, \mathcal{T}^A, \mathcal{R}^A)$ forms an MDP where $\mathcal{T}^A : \mathcal{S}^A \times \mathcal{A} \mapsto \Delta(\mathcal{S}^A)$ is the agent state transition function and $\mathcal{R}^A : \mathcal{S}^A \times \mathcal{A} \mapsto \mathbb{R}$ the agent state reward function such that:

$$\mathcal{T}^A(\boldsymbol{A}(t), \boldsymbol{a}(t)) = \sum_{\boldsymbol{o}(t+1) \in \Omega} f^A(\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)) \times P(\boldsymbol{o}(t+1)|\boldsymbol{a}(t), \boldsymbol{A}(t))$$

$$\mathcal{R}^A(\boldsymbol{A}(t), \boldsymbol{a}(t)) = \sum_{\boldsymbol{s}(t) \in \mathcal{S}} P(\boldsymbol{s}(t)|\boldsymbol{A}(t)) \mathcal{R}(\boldsymbol{s}(t), \boldsymbol{a}(t))$$

where:

$$P(\boldsymbol{o}(t+1)|\boldsymbol{a}(t), \boldsymbol{A}(t)) = \sum_{\boldsymbol{s} \in \mathcal{S}} \mathcal{O}(\boldsymbol{o}(t+1)|\boldsymbol{s}, \boldsymbol{a}(t)) \times \sum_{\boldsymbol{s} \in \mathcal{S}} \mathcal{T}(\boldsymbol{s}(t+1)|\boldsymbol{s}, \boldsymbol{a}(t))$$

Proof:

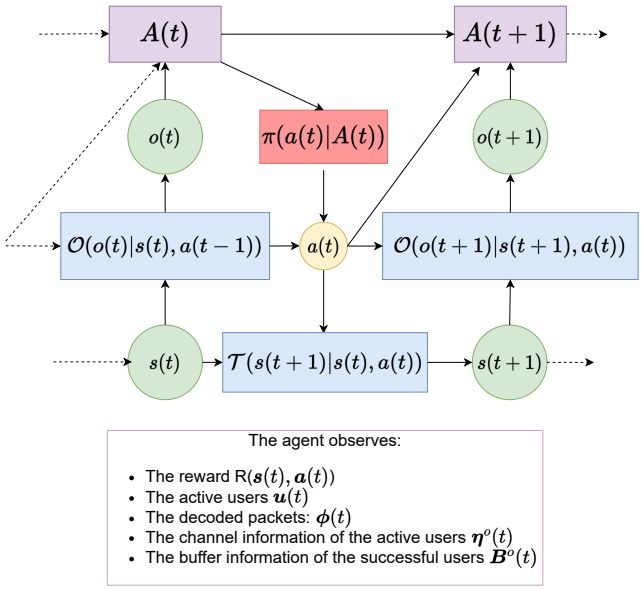


FIGURE 2: Formulation of the NOMA-URLLC problem.

The expressions of $\mathcal{T}^A$ and $\mathcal{R}^A$ are directly derived using the law of total probability and the Bayes formula. ■

The problem formulation is summarized in Fig. 2.

1) At the beginning of frame t, the system is at state $\boldsymbol{s}(t)$.
2) The agent can observe $\boldsymbol{o}(t) = \mathcal{O}(\boldsymbol{o}(t)|\boldsymbol{s}(t),\boldsymbol{a}(t-1))$.
3) It then computes the agent state using (16).
4) It makes an action $a(t) \sim \pi(\boldsymbol{a}(t)|\boldsymbol{A}(t))$.
5) The system then transitions to the next state as follows: $\boldsymbol{s}(t+1) \sim \mathcal{T}(\boldsymbol{s}(t+1)|\boldsymbol{s}(t),\boldsymbol{a}(t))$ and the agent receives: $R(\boldsymbol{s}(t),\boldsymbol{a}(t)), \boldsymbol{u}(t), \boldsymbol{\phi}(t), \boldsymbol{\eta}^o(t), \boldsymbol{B}^o(t)$.

Transforming the POMDP problem in an MDP allows us to leverage DRL algorithms in order to solve the optimization problem (P).

IV. Deep Reinforcement Learning Approach

A. Proximal Policy Optimization algorithm

The Proximal Policy Optimization (PPO) algorithm [31] is a Policy Gradient (PG) algorithm that benefits from the Trust Region Policy Optimization policy update [47] while being data efficient. The idea is to restrict the amplitude of the policy update in order to improve training stability while only using first-order optimization. PPO maximizes the following objective with respect to parameters θ:

$$\mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim(\pi_{\text{old}},\mathcal{T})}\left[\min\left(\frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})}A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a}), g(\nu)A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a})\right)\right] \tag{18}$$

with $g(\nu) = \text{clip}\left(\frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})}, 1-\nu, 1+\nu\right)$ and $\nu \in [0,1)$ a hyperparameter that indicates how far away the new policy can deviate from the old one. $A^{\pi_{\text{old}}}$ is the advantage function and is defined by:

$$A^{\pi_{\text{old}}}(\boldsymbol{s}(t),\boldsymbol{a}(t)) = Q^{\pi_{\text{old}}}(\boldsymbol{s}(t),\boldsymbol{a}(t)) - V^{\pi_{\text{old}}}(\boldsymbol{s}(t)) \tag{19}$$

It describes the value $Q^{\pi_{\text{old}}}(\boldsymbol{s}(t),\boldsymbol{a}(t))$ of an action $\boldsymbol{a}$ in a state $\boldsymbol{s}$ compared to the value of the state $V^{\pi_{\text{old}}}(\boldsymbol{s}(t))$ (how much better or worse it is to take this action). The estimated advantage function, noted $\hat{A}$ can be computed according to several methods that can be found in [48]. The most efficient method, which is also the one we adopt in NOMA-PPO, is the Generalized Advantage Estimation (GAE) algorithm [48]. This algorithm uses the temporal difference residuals $\delta^V(t) = r(t) - \gamma V(\boldsymbol{s}(t+1)) - V(\boldsymbol{s}(t))$ in order to define the *Generalized Advantage Estimator* $\hat{A}^{GAE}(t)$:

$$\hat{A}^{GAE}(t) = \sum_{l=0}^{\infty}(\gamma\lambda_{GAE})^l\delta^V(t+l) \tag{20}$$

where $\lambda_{GAE} \in [0,1]$ adjusts the bias-variance tradeoff. This method manages to reduce the variance of the gradient estimate and stabilizes training at the cost of introducing a bias. In practice, the value function V is approximated by a DNN with parameters φ: V_φ.

B. Exploiting Prior Knowledge

In our scheduling problem, the Earliest Deadline First (EDF) scheduler, which schedules pending packets in the increasing order of their deadline, intuitively is a good heuristic when the environment is fully observable by the scheduler. EDF is indeed known to be optimal in various deterministic [49] and stochastic (see e.g. [50]) settings. We thus adapt it to NOMA as follows: given the devices' buffers, $\boldsymbol{B}(t)$, EDF schedules the B users with the smallest head-of-line delay $d_k^h(t)$.

$$\text{EDF}(\boldsymbol{B}(t)) = (a_1,\ldots,a_K), \tag{21}$$
$$\text{where } a_k = \begin{cases} 1 & \text{if } k \in \arg_B\min(\{d_1^h(t),\ldots,d_K^h(t)\}) \\ 0 & \text{otherwise} \end{cases}$$

Note that in our POMDP problem, EDF cannot be implemented in practice, as it requires full observability of the system, but can serve as a valuable benchmark. We can further allow the scheduler to take into account the channel state, by introducing a prior regarding the channel quality. In particular, we define a prior on the channel f_{ch} as follows:

$$f_{\text{ch}}(\boldsymbol{\eta}(t),\boldsymbol{\tau}^{\boldsymbol{a}}) = (a_1,\ldots,a_K), \tag{22}$$
$$\text{where } a_k = \begin{cases} 0 & \text{if } \eta_k \le \eta^* \text{ and } \tau_k^a \le \tau^* \\ 1 & \text{otherwise} \end{cases}$$

where $\eta^* \ge 0$ and $\tau^* \ge 0$ are hyperparameters to determine the quality of a channel. Typically, η^* is the threshold that indicates when a user will not be decoded with a high probability, regardless of the others' channels and τ^* is the coherence time that indicates whether the last information we have on the channel is relevant or outdated. The intuition behind this prior is that a user should remain inactive if it experiences a very "bad" channel.

The resulting prior f is thus a combination of the EDF and channel prior:

$$f(\boldsymbol{a};\boldsymbol{A}) = EDF(\boldsymbol{B}(t)) \odot f_{\text{ch}}(\boldsymbol{\eta}(t), \boldsymbol{\tau}^{\boldsymbol{a}}) \tag{23}$$

In order to incorporate this prior knowledge into the RL agent, we introduce a Bayesian policy inspired by [32]. We express the posterior policy $q(\boldsymbol{a}|\boldsymbol{A};\theta_\pi)$ as a function of the prior over the agent state $f(\boldsymbol{a};\boldsymbol{A})$ and the task specific policy $\pi(\boldsymbol{a}|\boldsymbol{A};\theta_\pi)$ parameterized by θ_π with the Bayes rule:

$$q(\boldsymbol{a}|\boldsymbol{A};\theta_\pi) \propto \pi(\boldsymbol{a}|\boldsymbol{A};\theta_\pi) \odot f(\boldsymbol{a};\boldsymbol{A}) \tag{24}$$

C. Algorithm Overview and Architecture

The neural network architecture is described in Fig. 3. NOMA-PPO uses two neural networks, one for the policy and one for the critic. The input vector is the concatenation of the preprocessed buffer information $\boldsymbol{B}^A(t)$, the timing information $1/\boldsymbol{\tau}^p(t)$, $1/\boldsymbol{\tau}^a(t)$, $1/\boldsymbol{\tau}^s(t)$, the channel information $\boldsymbol{\eta}^A(t)$ and the last reward $r(t-1)$. Its size is thus $5K+1$.

Following a branching architecture, the policy network produces activation probabilities for each user, yielding K outputs: $\pi_\theta(\boldsymbol{a}|\boldsymbol{A}) = (\pi_\theta(a_1|\boldsymbol{A}), \pi_\theta(a_2|\boldsymbol{A}) \ldots, \pi_\theta(a_K|\boldsymbol{A}))$. Inspired by the BDQ architecture [24], which employs this approach for Q-learning, we handle the combinatorial action space in a manner that scales linearly with the number of users and adapt it to the PPO algorithm. These K outputs are then coordinated by a first block of hidden layers that are shared by all *branches*. This design balances the complexity of the model with the need to capture inter-dependencies among actions. While the single-layer branches simplify the model and enhance efficiency, the shared layers ensure that the critical inter-dependencies of our scheduling problem are captured.

On the other hand, the value network follows the same architecture of the policy network, except that it outputs a single value for the state value estimation. The procedure for training NOMA-PPO is developed in Algorithm 2. The training process for the algorithm consists of two phases: an initial offline training using synthetic data until satisfactory performance is achieved, followed by the deployment in the real environment where continuous updates of the policy and value network are performed in parallel, based on data collected during operation. Note that it is required to have a stationary environment across the initial training and the operation phases in order that the learning algorithm can efficiently exploit the training data. Moreover, as system parameters evolve or change, periodic maintenance or re-training of the algorithm is necessary, either on a scheduled basis or when a decline in performance is observed.

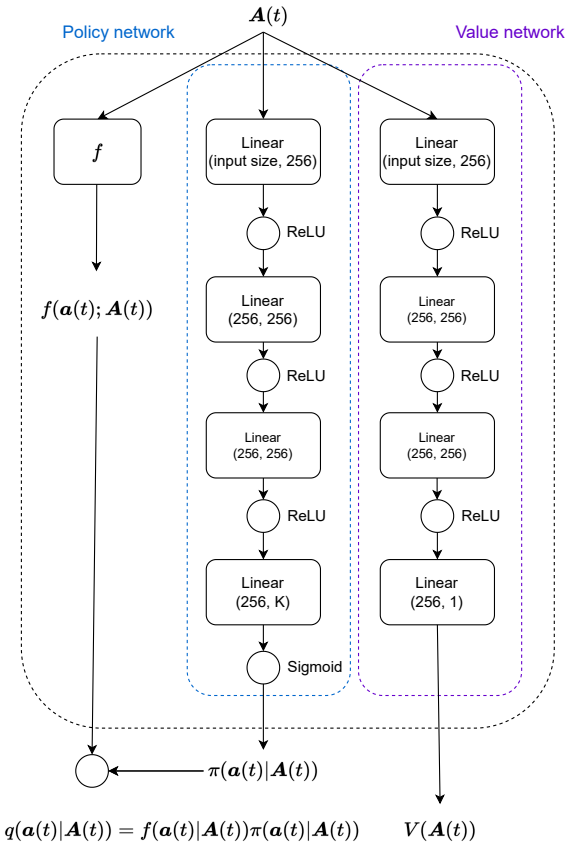


FIGURE 3: Architecture of the NOMA-PPO agent.

V. Experiments

A. Simulation Settings and Implementation Details

Simulations are conducted at the MAC layer. Our simulation settings (see Table 3) adopt the parameters of the factory automation use case of the 3GPP 5G NR specifications on URLLC [40] and industrial IoT [59]. Our radio frame is made of five time-slots ($T_f = 5T_s$), whose duration T_s is equivalent to an OFDM symbol in NR. It can be decomposed into an information part of duration T_i and a cyclic prefix of duration T_{cp}, which both depend on the subcarrier spacing Δf: $T_s = T_i + T_{cp}$ with $T_i = 1/\Delta f$. From the signal bandwidth we substract the subcarriers dedicated to uplink pilots, so that, when there are U polled devices and n_p pilots per device, the number of complex channel uses is $n = (W - n_p U \Delta f) T_i$ [37, Chapter 5]. The number of pilots per device can be obtained as follows: $n_p = \lceil W/W_c \rceil$, where $W_c = 1/(2T_d)$ [37, Chapter 2] is the coherence bandwidth and T_d is the delay spread.

Regarding the traffic model, we consider either a deterministic periodic traffic with period $1/\lambda$ or a probabilistic aperiodic traffic with average inter-arrival time $1/\lambda$. A packet can be decomposed into an information part of length L_i, a header part of length L_h, and a buffer description of length L_b, so that $L = L_i + L_h + L_b$. In URLLC, headers cannot indeed be neglected with respect to the message length. We assume that the information part, the header and the buffer information are jointly encoded [60]. The traffic parameters of Table 3 are taken from the factory automation use case of Release 16 [40]. On the downlink, the poll packet includes the vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_K) \in \{0,1\}^K$, whose size only

Algorithm 2: NOMA-PPO for URLLC uplink scheduling in NOMA systems.

Input: prior f, initial parameters of the policy network π_{θ_0} and the value network V_{φ_0};
for $j = 1, 2, \ldots, J$ **do**
  Run the posterior policy q_{θ_j} and collect a set of β trajectories $\{(\boldsymbol{A}_b(t), \pi_{\theta_j}(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t)), r_b(t))_{t=1,\ldots,T}\}_{b=1\ldots\beta}$.
  Compute the rewards-to-go $\hat{R}_b(t)$ for each trajectory: $\hat{R}_b(t) = \sum_{t'=t}^{T} \gamma^{t'} r_b(t')$
  Compute the values $V_{\phi_j}(\boldsymbol{A}_b(t))$ using the value network.
  Compute the advantage estimates $\hat{A}_b^{GAE}(t)$.
  Update the policy network by maximizing (18) with the Adam algorithm [51]:

$$\theta_{j+1} = \arg\max_{\theta} \frac{1}{\beta T} \left[\sum_{b=1}^{\beta} \sum_{t=1}^{T} \min \left(\frac{\pi_\theta(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t))}{\pi_{\theta_j}(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t))} \hat{A}_b^{GAE}(t), g(\nu) \hat{A}_b^{GAE}(t) \right) \right] \tag{25}$$

  Update the value network by minimizing the mean-squared error with the Adam algorithm:

$$\varphi_{j+1} = \arg\min_{\varphi} \frac{1}{\beta T} \sum_{b=1}^{\beta} \sum_{t=1}^{T} \left(V_\varphi(\boldsymbol{A}_b(t)) - \hat{R}_b(t) \right)^2 \tag{26}$$

TABLE 3: Network Simulation Settings.

Parameter	Notation	Value
Carrier frequency	f_c	4 GHz
Bandwidth[a]	W	38.16 MHz
Subcarrier spacing	Δf	30 kHz
Delay spread[b]	T_d	100 ns
OFDM symbol information part	T_i	33.33 μs
OFDM symbol cyclic prefix[c]	T_{cp}	2.34 μs
SIC limitation	B	3
Information length	L_i	32 bytes
Headers[d] length	L_h	46 bytes
Buffer information[e]	L_b	14 bytes
Average inter-arrival rate	$1/\lambda$	2 ms
Period in proba. periodic traffic	N_p	2 ms
Deadline	δ	1 ms
Network layout	$\ell \times \ell'$	50×120 m^2
Noise Power Spectral Density	N_0	-174 dBm/Hz
BS noise figure	N_F	5 dB
BS antenna height	$\tilde{h}_b$	3 m
BS antenna gain	G_b	5 dBi
BS number of antennas	n_a	4
Device transmit power	p	23 dBm
Device antenna height	$\tilde{h}_d$	1.5 m
Device antenna gain	G_d	0 dBi
Device speed	v	3 km/h

[a] For a channel bandwidth of 40 MHz, the signal occupies 38.16 MHz after having excluded guard bands [52].
[b] Typical delay spread for an indoor hot-spot scenario with carrier frequency 4 GHz [40].
[c] Normal cyclic prefix duration for symbols not at the start or in the middle of the subframe [53].
[e] Size of an array of size 56 corresponding to a maximum deadline of 56 frames, i.e., 10 ms, with 3 bits entries giving the number of packets for every deadline.
[d] Headers include 2 bytes of CRC [54], 1 byte for MAC [5], 0 byte for RLC [55], 2 bytes for PDCP [56], 0 byte for SDAP [57] and 40 bytes for IPv6 [58].

grows linearly with the number of devices K and remains reasonable given the number of considered devices and the bandwidth W available on the downlink. In the numerical experiments, we consider up to $K = 40$ devices for a BS, which is consistent with 3GPP URLLC performance evaluations [40].

For realistic numerical experiments, we partly adopt the scenario proposed in [40, Table A.2.2-1] for the factory automation use case, with a single BS. The network layout is a rectangle of size $\ell \times \ell'$; the BS is positioned at its center at a height $\tilde{h}_b$ and serves devices, each at height $\tilde{h}_d$ and moving with velocity v. Devices are uniformly distributed within the network area. Devices and BS benefit from antenna gains G_b and G_d respectively. For a speed of $v = 3$ km/h, we obtain a coherence time of $T_c = c/(8 f_c v) = 11.2$ ms, which corresponds to 63 radio frames. We choose an episode length of 200 frames that allows us to consider speeds below 1 km/h. The path-loss model is the ITU InH NLOS [61]. In such an environment, the correlation distance is between a quarter and three quarters of the wavelength [62], i.e., between 1.8 and 5.6 cm in our scenario. It is thus possible to obtain independence between the antennas by spacing them a few centimeters apart. The BS has a noise figure N_F, so that the noise power is $\sigma_n^2 = N_0 W N_F$, where N_0 is the noise power spectral density. Typical values for the channel parameters are given in Table 3. We express the deadlines and inter-arrival time in term of frames. Indeed, given the frame duration T_f, we can deduce that the average inter-arrival time of 2 ms corresponds to 11.2 frames and that the deadline of 1 ms to 5.6 frames.

The parameters of the DRL algorithms are given in Table 4. We preprocess the agent state as follows. In order to reduce the dimension of the buffer information, the matrix $\boldsymbol{B}^A(t)$ is transformed into a vector of size K of head-of-line delays for each agent. In order to improve stability of speed up training, we normalize $\boldsymbol{\tau}^p, \boldsymbol{\tau}^a, \boldsymbol{\tau}^s$ between 0 and 1 by taking $1/\boldsymbol{\tau}^p, 1/\boldsymbol{\tau}^a, 1/\boldsymbol{\tau}^s$. The channel threshold η^* is calculated using (8) such that the error probability in absence of interference corresponding to η^* is equal to 10^{-5}. Finally, note that the history length $K_{\hbar}$ scales in our experiments with the number of devices K. This choice

TABLE 4: Parameters of the DRL algorithms.

Parameter	Value
Input size (H_{in})	$5K+1$
Hidden size (H)	256
Discount factor (γ)	0.3
Learning rate actor	10^{-4}
Learning rate critic	10^{-3}
Batch size	128
History length ($K_\hbar$)	K
Episode length (T)	200 slots
Training length (J)	10k episodes
Activation functions	ReLU
Number of seeds	5
λ_{GAE}	0.95

has been experimentally found to provide a good tradeoff between computational efficiency and model performance.

URLLC score

In order to compare our algorithm to the traditional benchmarks, we define the *URLLC score* as the number of successfully transmitted packets over the number of received packets. In the following experiments, the URLLC score is computed over 500 episodes which corresponds to approximately $2 \cdot 10^5$ generated packets according to the traffic parameters in Table 3. Therefore, a URLLC score of 1 means that the reliability is greater than $1 - 10^5$.

B. Benchmarks

For all baselines, when the BS receives two or more packets at the same time, we use the SIC procedure described in Section B to decode the packets.

- **Random Scheduler**: This scheduler schedules a subset of B devices uniformly at random.
- **EDF**: This scheduler schedules pending packets in the increasing order of their deadlines, see (21). Again, it cannot be implemented in practice on the uplink because of the assumed full observability of the device buffers.
- **SA-NOMA-SIC**: This baseline is a grant-free approach that follows the work of [9]. It combines SA with SIC. At each frame, devices transmit their packet with the same probability p. Regarding re-transmissions, we use the *proactive* scheme [8]: a user can re-transmit the same packet with probability p until it is delivered or expired. The probability p is empirically optimized such that the URLLC score is maximized for every scenario.
- **RDQN-NOMA Scheduler**: The standard DQN algorithm proposed by [27] is the traditional approach to solve POMDP problems. The idea is to use an RNN to handle partial observability. We directly apply this algorithm in order to solve (P). The action space of the RL agent is the set of combinations of B or more devices to poll.
- **Branching DQN (BDQ)**: this baseline is a version of the Dueling Double DQN algorithm from [24] that uses a branching architecture in order to handle a combinatorial action space.
- **iDRQN-NOMA**: This baseline is a fully distributed Multi-Agent Reinforcement Learning (MARL) algorithm for grant-free multiple access that follows the solution of [15] where each device is modeled by a Deep Q-network and decides to access the medium based on its local information: the state of its buffer and its channel state. This baseline uses a RNN, a Gated Recurrent Unit (GRU) layer [63] in particular, as it is a standard approach to tackle partial observability. Additionally, we extend the work of [15] to NOMA systems by adapting the reward function as follows: at the end of every frame t, every user k receives the same reward:

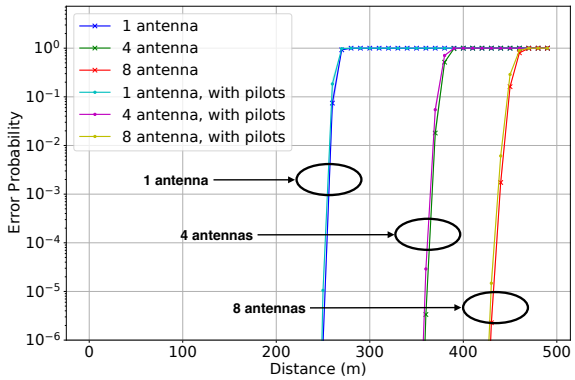
$$R^k(s_k(t), a_k(t)) = \begin{cases} \sum_{i \in \mathcal{U}(t)} \phi_i(t) & \text{if } |\mathcal{U}(t)| \leq B \\ -1 & \text{otherwise} \end{cases} \tag{27}$$

- **NOMA-PPO-no-prior**: This baseline is the proposed approach, however without using prior information over the agent state.
- **NOMA-PPO-no-agent-state**: this approach is our NOMA-PPO algorithm without the agent state. It deals with partial observability using the action observation history as the input to the policy. It then processes it using a recurrent neural network (RNN) [27].

In order to be fair in the experiments, we modify the frame structure of the two grant-free approaches SA-NOMA-SIC and iDRQN-NOMA and divide it into four time-slots of duration T_s: an uplink transmission symbol, a guard symbol, a downlink ACK/NACK and a guard symbol.

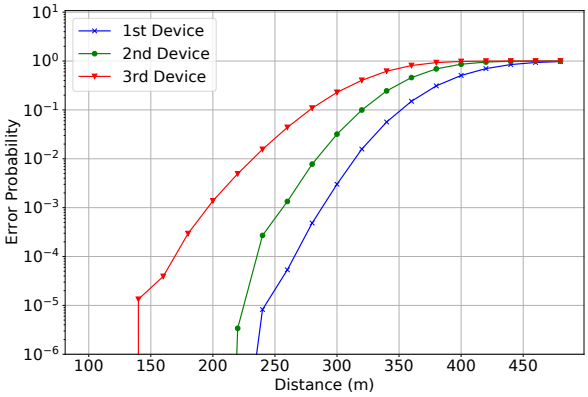
C. Study of the Channel Model

In this section, we study the behavior of the channel. In Fig. 4a, we show the channel error probability ϵ as a function of the distance between a device and the BS, involved in a point-to-point transmission without interference. Results are shown for different number of antennas at the BS and with or without the pilot signals. We see that there are roughly three regimes that can be distinguished. When the distance is small, the error probability is very small (less than 10^{-6}). When the distance to the BS is too large, the error probability is close to 1. In this regime, there is no hope to guarantee URLLC requirements. In an intermediate regime that depends on the number of antennas and the number of decoded devices, the error probability is not negligible but the URLLC requirements could be met with an appropriate scheduling. In this case, the SINR model is required to benefit from the channel evolution for every device. As expected, increasing the number of antennas at the



(a) For a single device ($K = 1$)

.



(b) For three devices after SIC ($K = 3$)

.

FIGURE 4: Packet error probability ϵ as a function of the distance to the BS.

BS improves the reliability. At last, reserving some resource for pilots has a negligible influence on the performance.

In Fig. 4b, we show the error probability as a function of the distance of the three devices from the BS. The three devices transmit simultaneously to the BS, which performs SIC. The resulting error probabilities are shown for the first, second and third decoded signals respectively. We again observe the three regimes, however with an offset according to the rank of decoding. The intermediate regime ranges here approximately between 150 m and 300 m.

D. Convergence Analysis

In Fig. 5a, we show the evolution of the URLLC score during the training of 18 learning agents under the probabilistic aperiodic traffic. First, we can see that NOMA-PPO converges the fastest and with the smallest variance to its asymptotic value. Second, we see that not only does the prior help NOMA-PPO reach a better optimum, it also increases the convergence speed and reduces the variance. Third, we can observe that NOMA-PPO-no-prior's performance closely aligns with NOMA-PPO-no-agent-state. This suggests that utilizing the agent state achieves the same benefits as managing partial observability with a RNN, but with reduced complexity. However, we can see that training NOMA-PPO-no-agent-state is longer, primarly due to the higher number of parameters in the GRU layer.

Fourth, while the MARL grant-free approach, iDRQN-NOMA, reaches the second best optimum in terms of URLLC score, it converges slower than NOMA-PPO and with a larger variance. This can be accounted for by the fact that agents must coordinate independently, solely using the BS's feedback. Furthermore, we observe that the DRQN-NOMA scheduler does not manage to converge due to the combinatorial action space. Indeed, there are $2^K - \sum_{k=0}^{B-1} \binom{K}{k} = 261,972$ possible actions for $K = 18$ and $B = 3$, thus choosing the appropriate action is challenging. In light of the lack of convergence of this algorithm, we exclude it from future experimental baselines. Finally, we observe that the BDQ algorithm comes third in term of asymptotic URLLC score but suffers from high variance. We notice that for the DRQN-NOMA scheduler and iDRQN-NOMA, the score does not evolve in the first thousand episodes. It is because of the "warm up" stage where we collect trajectories in order to fill the replay buffer of the agents without updating them.

E. Performance in the 3GPP Scenario

In Fig. 5, we study the performance of our algorithm in the 3GPP scenarios with two different traffic models. On the one hand, we study in Fig. 5b the evolution of the URLLC score as a function of the number of devices on the deterministic periodic traffic and on the other hand, the evolution of the URLLC score and the Jain's Index on the probabilistic aperiodic traffic in Fig. 5c and Fig. 5d respectively. The Jain's index is computed with the URLLC scores.

We observe that our approach, NOMA-PPO, consistently outperforms all benchmarks in URLLC score and fairness across various scenarios, with the exception of the EDF scheduler, which benefits from full observability over devices' buffers. This superior performance can be explained by our technical contributions to better handle partial observability and the combinatorial action space, augmented with prior knowledge.

Indeed, we first observe that our proposed solution surpasses the BDQ algorithm, particularly in high-density device scenarios, where it often converges to suboptimal policies. While BDQ can handle large action spaces, its inability to effectively manage partial observability leads to the convergence to suboptimal policies as the number of users increases. In contrast, NOMA-PPO successfully mitigates the issues of partial observability thanks to the integration of the agent state.

Regarding grant-free methods, we observe distinct behaviors under different traffic patterns. For instance, the iDRQN-NOMA algorithm struggles to converge with 30 devices and

more in deterministic periodic traffic but outperforms SA-NOMA-SIC for up to 18 users in probabilistic aperiodic traffic. Beyond 18 devices, the performance deteriorates as coordinating a high number of users becomes challenging. This complexity leads to instabilities inherent to independent learning, that suffers from non-stationarity arising from concurrent learning processes. In comparison, NOMA-PPO offers a central control mechanism for transmissions. This is advantageous over distributed methods, which see the number of collisions increasing with a growing number of devices. Additionally, its capability to incorporate prior information about the wireless system enables the discovery of more effective policies in scenarios with an increasing number of users. This feature ensures not only an efficient scheduling of devices with favorable channel conditions but also prioritization of packets close to their deadlines.

Finally, it is important to note that the DRL baselines, despite their potential, often get trapped in local optima, leading to suboptimal policies that even underperform compared to a random scheduler when the number of users increases.

F. Performance in Different Channel Conditions

In this subsection, we analyze the behavior of NOMA-PPO under diverse channel conditions.

Fig. 6 shows the evolution of the URLLC score during the training as a function of the number of iterations in the probabilistic aperiodic traffic of the 3GPP scenario where parameters are listed in Table. 3, for 10 devices. Besides, we test two different values for the deadline-coherence time ratio where packets have a deadline of 10 ms, a coherence time of 1.4 ms (Fig. 6a) and 0.34 ms (Fig. 6b).

We compare the NOMA-PPO agent with:

- **NOMA-PPO (full CSI)**: the version of NOMA-PPO where the channel information of all users is observable.
- **NOMA-PPO (no CSI)**: the version of the algorithm where we remove the channel information from the agent state.

First, we can see on all figures that the complete observability of the users' channels enriches the agent state and results in superior performance compared to the versions lacking this feature. Second, as depicted in Fig. 6a, when the coherence time is long enough, NOMA-PPO manages to accurately estimate the true channel based on the agent state. This enables it to reach similar performance to NOMA-PPO with full CSI. Third, we observe in Fig. 6b, when the coherence time is too short, that NOMA-PPO fails to reach the performance of NOMA-PPO with full observability but still outperforms the one with no CSI. The reason is that the coherence time must be large enough compared to the deadline so that the algorithm has time to both sense the channel and schedule the packet when the channel conditions are favorable. Furthermore, we notice that NOMA-PPO outperforms the EDF scheduler in Figure 6(a). The reason for this difference stems from the inherent design of EDF scheduler which focuses on serving packets based purely on their deadlines, neglecting the variability in channel conditions. Indeed, we observe that in Figure 6(a), where the coherence time is longer, the channel remains relatively stable over several frames, which enables NOMA-PPO to exploit the channel conditions, thus demonstrating superior performance over EDF. In contrast, when the coherence time is short, as in Figure 6(b), the rapidly fluctuating channel conditions render channel-based decisions less predictable and less exploitable. In such scenarios, the straightforward deadline-driven approach of EDF showcases robust performance.

Finally, we noted through simulations not depicted here, that when the number of users is too small, NOMA-PPO (no CSI) attains equivalent performance to both NOMA-PPO (full CSI) and NOMA-PPO. This can be explained by a multi-user diversity gain that increases with the number of devices.

TABLE 5: FLOPs for the Deep Neural Networks.

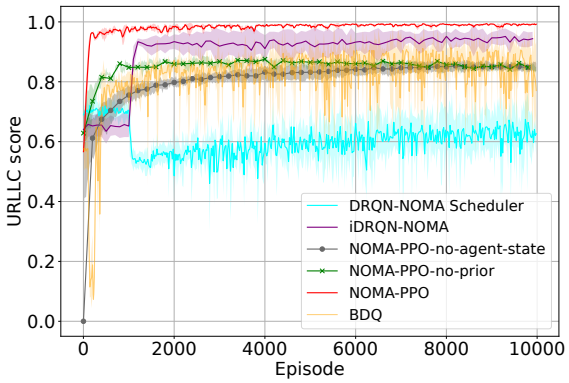
Algorithm	FLOPs
NOMA-PPO	$3,072K + 263,424$
BDQ	$4,096K + 394,496$
iDRQN-NOMA (1 agent)	$406,528K$

G. Complexity Analysis

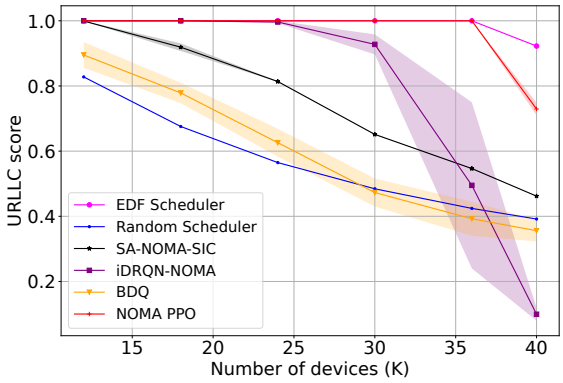
We evaluate the complexity of our Deep Learning architecture in terms of Floating Point Operations (FLOPs) that occur during a single forward pass of the neural network. Given that a connection between 2 neurons involves 2 operations (a multiplication and an addition) the FLOPs of a linear layer operation is $2 \times$ input size $\times$ output size. The GRU layer is made of four operations: the reset gate, the update gate, the candidate hidden state and the new gate [63], each being made of matrix multiplications, additions, and activation functions. Let H_{in} the size of the input and H the size of the hidden layer. We can express the FLOPs of each unit:

- The reset and update gate have the same structure: 2 matrix multiplications, 3 additions and 1 sigmoid activation. The resulting number of FLOPs for both gates is thus: $4H(H_{in} + H) + 6H$.
- The new gate contribution includes 2 matrix multiplications, a Hadamard product, a $tanh$ activation and the addition of the bias terms: $2H(H_{in} + H) + 4H$.
- Finally, the new hidden state involves two Hadamard products, and two additions: $4H$.

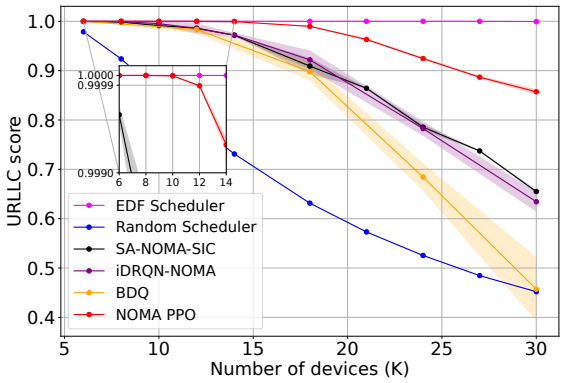
In addition, the complexity of the GRU depends on the size of the history. In our problem, we set the history size to the number of users K. In total, the number of FLOPs of a GRU layer is $6HK(H_{in} + H) + 10HK$.



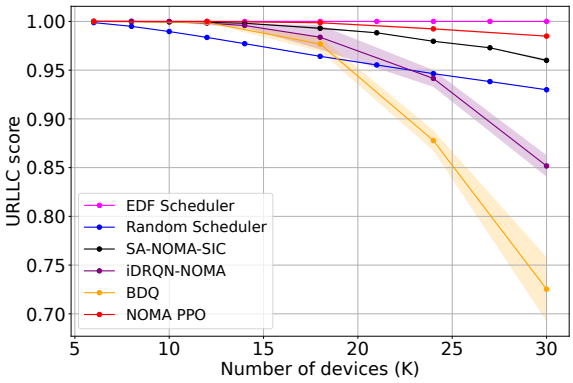
(a) Evolution of the URLLC score during training for $K = 18$ users.



(b) URLLC score in the 3GPP deterministic periodic scenario.



(c) URLLC score in the 3GPP probabilistic aperiodic scenario.



(d) Jain's fairness index in the 3GPP probabilistic aperiodic scenario.

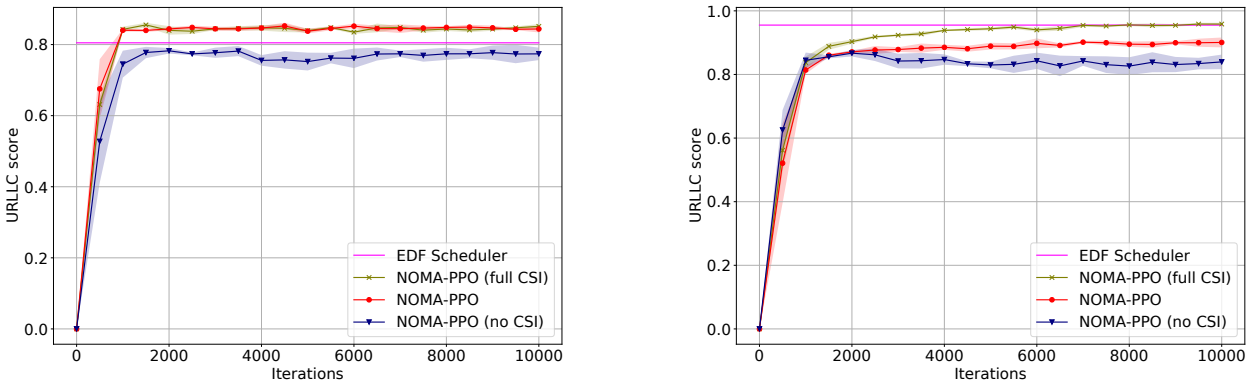
FIGURE 5: Performance metrics in the 3GPP scenario.

The number of FLOPs of the learning algorithms are given in Table 5 (using the numerical values of Table 4). First, we can see that all algorithms have a linear complexity in the number of users and differ by their slope. Second, the BDQ algorithm has a complexity greater than NOMA-PPO due to the use of an advantage and a value network during inference. Third, the complexity of one iDRQN agent is larger than the complexity of the centralized approach we propose due to the use of a GRU layer to process the action-observation history.

VI. Conclusion

In this paper, we propose a novel approach for satisfying Ultra Reliable Low Latency Communications (URLLC) requirements and strict deadlines in IoT networks, employing Non-Orthogonal Multiple Access (NOMA) for uplink communications.

Our proposed approach, NOMA-PPO, addresses the challenges posed by the NOMA uplink URLLC scheduling problem, namely the combinatorial action space and the partial observability. NOMA-PPO tackles these challenges by bringing three technical contributions. First, it formulates the NOMA-URLLC problem as a Partially Observable Markov Decision Process (POMDP), and introduces the concept of *agent state*, as sufficient statistic for the past actions and observations. This reformulation allows us to extend the state-of-the-art Proximal Policy Optimization (PPO) algorithm to handle a combinatorial action space thanks to a branching policy network. Finally, NOMA-PPO is able to incorporate prior knowledge over the system into the learning algorithm by employing a Bayesian policy. We demonstrate that our approach outperforms traditional Multiple Access and Deep Reinforcement Learning benchmarks in 3GPP scenarios for different traffic models (probabilistic aperiodic and deterministic periodic) in terms of URLLC score, fairness, and convergence speed. Finally, we show that our algorithm is robust under diverse channel configurations and is capable to leverage channel information. As our algorithm does not depend on the channel model assumption (it is model-free) and learns from the interactions with the



(a) $T_c = 1.4$ms, $K = 10$ users.

(b) $T_c = 0.34$ms, $K = 10$ users.

FIGURE 6: Evolution of the URLLC score during training under different channel conditions.

channel, we can easily extend the results to more complex channel models (e.g. with channel estimation errors). Future work can involve the implementation of NOMA-PPO in a real-world environment system to fully assess its practical efficacy.

Appendix A
Proof of Proposition 3.1

Let $\boldsymbol{s(t)} = \langle \boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t) \rangle$. We need to prove that $\boldsymbol{A(t)}$ is a sufficient statistic for the history $\hbar(t)$ in order to predict $\boldsymbol{s(t)}$ i.e.:

$$P(\boldsymbol{s(t)}|\hbar(t)) = P(\boldsymbol{s(t)}|\boldsymbol{A(t)}) \tag{28}$$

According to the Bayes rule, we have:

$$\begin{aligned} P(\boldsymbol{s}(t)|\hbar(t)) &= P(\boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t)|\hbar(t)) \\ &= P(\boldsymbol{B}(t)|\boldsymbol{\eta}(t), \boldsymbol{o}(t), \hbar_t) \\ &\quad \times P(\boldsymbol{\eta}(t)|\boldsymbol{o}(t), \hbar(t)) \times P(\boldsymbol{o}(t)|\hbar(t)) \end{aligned}$$

Besides, as $\hbar(t) = \{a(0), o(1), a(1), ..., a(t-1), o(t)\}$ and $\boldsymbol{B}(t)$ is independent of $\boldsymbol{\eta}(t)$, we have:

$$P(\boldsymbol{s}(t)|\hbar(t)) = P(\boldsymbol{B}(t)|\hbar(t))P(\boldsymbol{\eta}(t)|\hbar(t))$$

Each channel is independent so let's compute $P(\eta_k(t)|\hbar(t))$. For a device k, $\eta_k(t)$ is conditionally independent of $\boldsymbol{a}_k(t), \boldsymbol{b}_k^o(t)$ and $r(t-1)$ given $\eta_k^o(t)$. Thus, we can write:

$$\begin{aligned} P(\eta_k(t)|\hbar(t)) &= P(\eta_k(t)|\eta_k^o(t-1), \dots, \eta_k^o(0)) \\ &\overset{(a)}{=} P(\eta_k(t)|u_k(t-1)\eta_k(t-1), \dots, u_k(0)\eta_k(0)) \\ &\overset{(b)}{=} P(\eta_k(t)|\eta_k(\tau_k^a(t)), \tau_k^a(t)) \\ &\overset{(c)}{=} P(\eta_k(t)|\boldsymbol{A}(t)) \end{aligned}$$

where (a) comes from the definition of η_k^o (see Section B-2); (b) comes from the fact that the channel realizations are Markovian and that $\eta_k(\tau_k^a(t))$ is the last observed channel realization for the device k; (c) results from the conditional independence of $\eta_k(t)$ from $\boldsymbol{B}^A(t), \boldsymbol{\tau}^a(t), \boldsymbol{\tau}^s(t), r(t-1)$ given $\eta_k^A(t)$ and $\tau_k^a(t)$.

Finally, as each user's buffer is independent, we are going to prove by induction that $P(\boldsymbol{b}_k(t)|\hbar(t)) = P(\boldsymbol{b}_k(t)|\boldsymbol{A}(t)), \forall k, \forall t \geq 0$. First, $P(\boldsymbol{s}(0)|\hbar(0)) = P(s(0)|\ o(0)) = P(s(0)|\boldsymbol{A}(0))$. Let $t \geq 0$ and $k \in [1, K]$. Let's assume that $P(\boldsymbol{b}_k(t)|\hbar(t)) = P(\boldsymbol{b}_k(t)|\boldsymbol{A}(t))$. By definition of the agent state (see Definition 2) and as $\boldsymbol{b}_k(t+1)$ only depends on $\boldsymbol{b}_k^A(t+1)$, we have:

$$P(\boldsymbol{b}_k(t+1)|\boldsymbol{A}(t+1)) = P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) \tag{29}$$

$$\boldsymbol{b}_k^A(t+1) = \begin{cases} \boldsymbol{b}_k^o(t+1) & \text{if } \phi_k(t) = 1 \\ \boldsymbol{b}_k^A(t) & \text{if } \phi_k(t) = 0 \end{cases} \tag{30}$$

Therefore,

$$P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) = \begin{cases} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^o(t+1)), & \text{if } \phi_k(t) = 1, \\ P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t)), & \text{if } \phi_k(t) = 0, \end{cases} \tag{31}$$

Besides, as $\boldsymbol{b}_k(t+1)$ is conditionally independent of $r(t)$, $\boldsymbol{H}^o(t+1), \boldsymbol{u}(t)$ given $\boldsymbol{\phi}(t), \boldsymbol{b}_k^o(t+1)$ we can write $P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^o(t+1)) = P(\boldsymbol{b}_k(t+1)|\boldsymbol{o}(t+1))$. Therefore:

$$\begin{aligned} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) &\overset{(c)}{=} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t), \boldsymbol{o}(t+1)) \\ &\overset{(d)}{=} P(\boldsymbol{b}_k(t+1)|\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)) \\ &\overset{(e)}{=} P(\boldsymbol{b}_k(t+1)|\hbar(t+1)) \end{aligned}$$

where (c) comes from merging the equations in (31) and noticing that $\phi_k(t) \in \boldsymbol{o}(t+1)$; (d) comes from the fact that $\boldsymbol{b}_k(t+1)$ only depends on $\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)$ through $\boldsymbol{b}_k^A(t)$ and $\boldsymbol{o}(t+1)$. Finally, (e) comes from the induction hypothesis.

REFERENCES

[1] "Study on scenarios and requirements for next generation access technologies," 3rd Generation Partnership Project (3GPP), TR 38.913. [Online]. Available: http://www.3gpp.org/DynaReport/38913.htm

[2] Y. Saito, Y. Kishiyama, A. Benjebbour, T. Nakamura, A. Li, and K. Higuchi, "Non-orthogonal multiple access (NOMA) for cellular future radio access," in *2013 IEEE 77th Vehicular Technology Conf. (VTC Spring)*, 2013, pp. 1–5.

[3] G. Cuozzo *et al.*, "Enabling URLLC in 5G NR IIoT networks: A full-stack end-to-end analysis," in *2022 Joint Eur. Conf. on Netw. and Commun. 6G Summit (EuCNC/6G Summit)*, 2022, pp. 333–338.

[4] M. W. Nomeir, Y. Gadallah, and K. G. Seddik, "Uplink scheduling for mixed grant-based eMBB and grant-free URLLC traffic in 5G networks," in *2021 17th Int. Conf. on Wireless and Mobile Comput. Netw. and Commun. (WiMob)*, 2021, pp. 187–192.

[5] "NR; Medium Access Control (MAC) protocol specification," 3rd Generation Partnership Project (3GPP), TS 38.321. [Online]. Available: http://www.3gpp.org/DynaReport/38321.htm

[6] Y. Feng, A. Nirmalathas, and E. Wong, "A predictive semi-persistent scheduling scheme for low-latency applications in LTE and NR networks," in *2019 IEEE Int. Conf. on Commun. (ICC)*, 2019, pp. 1–6.

[7] S. E. Elayoubi, P. Brown, M. Deghel, and A. Galindo-Serrano, "Radio resource allocation and retransmission schemes for URLLC over 5G networks," *IEEE J. Sel. Areas in Commun.*, vol. 37, no. 4, pp. 896–904, 2019.

[8] N. H. Mahmood, R. Abreu, R. Böhnke, M. Schubert, G. Berardinelli, and T. H. Jacobsen, "Uplink grant-free access solutions for URLLC services in 5G new radio," in *2019 16th Int. Symp. on Wireless Commun. Syst. (ISWCS)*, 2019, pp. 607–612.

[9] S. A. Tegos, P. D. Diamantoulakis, A. S. Lioumpas, P. G. Sarigiannidis, and G. K. Karagiannidis, "Slotted ALOHA With NOMA for the next generation IoT," *IEEE Trans. on Commun.*, vol. 68, no. 10, pp. 6289–6301, 2020.

[10] M. B. Shahab, R. Abbas, M. Shirvanimoghaddam, and S. J. Johnson, "Grant-free non-orthogonal multiple access for IoT: A survey," *IEEE Commun. Surveys Tutorials*, vol. 22, no. 3, pp. 1805–1838, 2020.

[11] Y. Liu, Y. Deng, M. Elkashlan, A. Nallanathan, and G. K. Karagiannidis, "Analyzing grant-free access for URLLC service," *IEEE J. Sel. Areas in Commun.*, vol. 39, no. 3, pp. 741–755, 2021.

[12] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[13] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep reinforcement learning for internet of things: A comprehensive survey," *IEEE Commun. Surveys Tutorials*, vol. 23, no. 3, pp. 1659–1692, 2021.

[14] H.-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, and L. Liu, "Distributive dynamic spectrum access through deep reinforcement learning: A reservoir computing-based approach," *IEEE Internet of Things J.*, vol. 6, no. 2, pp. 1938–1948, 2019.

[15] Y. Xu, J. Yu, and R. M. Buehrer, "The application of deep reinforcement learning to distributed spectrum access in dynamic heterogeneous environments with partial observations," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4494–4506, 2020.

[16] X. Tan *et al.*, "Cooperative multi-agent reinforcement-learning-based distributed dynamic spectrum access in cognitive radio networks," *IEEE Internet of Things J.*, vol. 9, no. 19, pp. 19 477–19 488, 2022.

[17] H. Yang, Z. Xiong, J. Zhao, D. Niyato, C. Yuen, and R. Deng, "Deep reinforcement learning based massive access management for ultra-reliable low-latency communications," *IEEE Trans. on Wireless Commun.*, vol. 20, no. 5, pp. 2977–2990, 2021.

[18] W. Ahsan, W. Yi, Y. Liu, and A. Nallanathan, "A reliable reinforcement learning for resource allocation in uplink NOMA-URLLC networks," *IEEE Trans. on Wireless Commun.*, vol. 21, no. 8, pp. 5989–6002, 2022.

[19] Y. Liu, Y. Deng, H. Zhou, M. Elkashlan, and A. Nallanathan, "A general deep reinforcement learning framework for grant-free NOMA optimization in mURLLC," *arXiv*, no. 2101.00515, 2021.

[20] B.-M. Robaglia, A. Destounis, M. Coupechoux, and D. Tsilimantos, "Deep reinforcement learning for scheduling uplink iot traffic with strict deadlines," in *2021 IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6.

[21] G. Dulac-Arnold *et al.*, "Deep reinforcement learning in large discrete action spaces," *arXiv*, no. 1512.07679, 2015.

[22] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv*, no. 1509.02971, 2015.

[23] L. Metz, J. Ibarz, N. Jaitly, and J. Davidson, "Discrete sequential prediction of continuous actions for deep RL," *arXiv*, no. 1705.05035, 2017.

[24] A. Tavakoli, F. Pardo, and P. Kormushev, "Action branching architectures for deep reinforcement learning," *Proc. AAAI Conf. on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[25] E. J. Sondik, "The optimal control of partially observable markov processes," Ph.D. dissertation, Stanford University, 1971.

[26] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[27] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI fall symposium series*, 2015, pp. 29–37.

[28] M. Igl, L. Zintgraf, T. Le, F. Wood, and S. Whiteson, "Deep variational reinforcement learning for POMDPs," in *Proc. Int. Conf. on Machine Learning*, no. PMLR:80, 2018, pp. 2117–2126.

[29] J. Subramanian, A. Sinha, R. Seraj, and A. Mahajan, "Approximate information state for approximate planning and reinforcement learning in partially observed systems," *J. Mach. Learn. Res.*, vol. 23, no. 12, pp. 1–83, 2022.

[30] M. A. Wiering and M. Van Otterlo, Eds., *Reinforcement learning - state-of-the-art*. Berlin-Heidelberg: Springer, 2012.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, no. 1707.06347, 2017.

[32] M. K. Titsias and S. Nikoloutsopoulos, "Bayesian transfer reinforcement learning with prior knowledge rules," *arXiv*, no. 1810.00468, 2018.

[33] L. Salaün, M. Coupechoux, and C. S. Chen, "Joint subcarrier and power allocation in noma: Optimal and approximate algorithms," *IEEE Trans. on Signal Proc.*, vol. 68, pp. 2215–2230, 2020.

[34] H. Ren, C. Pan, Y. Deng, M. Elkashlan, and A. Nallanathan, "Joint power and blocklength optimization for URLLC in a factory automation scenario," *IEEE Trans. on Wireless Commun.*, vol. 19, no. 3, pp. 1786–1801, 2020.

[35] M. Kobayashi and G. Caire, "Joint beamforming and scheduling for a multi-antenna downlink with imperfect transmitter channel knowledge," *IEEE J. Sel. Areas in Commun.*, vol. 25, no. 7, pp. 1468–1477, 2007.

[36] W. C. Jakes and D. C. Cox, *Microwave mobile communications*. Hoboken: Wiley-IEEE press, 1994.

[37] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. New-York: Cambridge university press, 2005.

[38] A. Goldsmith, *Wireless communications*. New-York: Cambridge university press, 2005.

[39] Y. Tokgoz and B. Rao, "The effect of imperfect channel estimation on the performance of maximum ratio combining in the presence of cochannel interference," *IEEE Trans. on Vehicular Tech.*, vol. 55, no. 5, pp. 1527–1534, 2006.

[40] "Study on physical layer enhancements for NR ultra-reliable and low latency case (URLLC)," 3rd Generation Partnership Project (3GPP), TR 38.824. [Online]. Available: http://www.3gpp.org/DynaReport/38824.htm

[41] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. on Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

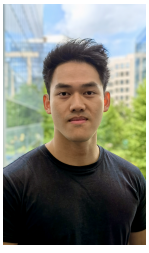
[42] I.-H. Hou and P. R. Kumar, *Packets with deadlines: A framework for real-time wireless networks*. Berlin-Heidelberg: Springer, 2013.

[43] "Technical Specification Group Services and System Aspects; Study on Communication for Automation in Vertical domains (CAV)," 3rd Generation Partnership Project (3GPP), TS 22.804. [Online]. Available: http://www.3gpp.org/DynaReport/22804.htm

[44] "Feasibility Study on Licensed-Assisted Access to Unlicensed Spectrum," 3rd Generation Partnership Project (3GPP), TR 36.889. [Online]. Available: http://www.3gpp.org/DynaReport/36889.htm

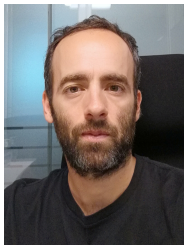
[45] Z. Guo, Z. Chen, P. Liu, J. Luo, X. Yang, and X. Sun, "Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks," *IEEE J. Sel. Areas in Commun.*, vol. 40, no. 5, pp. 1587–1599, 2022.

[46] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Math. of oper. res.*, vol. 12, no. 3, pp. 441–450, 1987.
[47] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Int. conf. on machine learning*, no. PMLR:37, 2015, pp. 1889–1897.
[48] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv*, no. 1506.02438, 2015.
[49] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. Buttazzo, *Deadline scheduling for real-time systems: EDF and related algorithms*. Bost, Dortrecht, London: Kluwer Academic Publishers, 1998.
[50] P. Moyal, "On queues with impatience: stability, and the optimality of earliest deadline first," *Queueing Systems*, vol. 75, pp. 211–242, 2013.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv*, no. 1412.6980, 2014.
[52] "NR; Base Station (BS) radio transmission and reception," 3rd Generation Partnership Project (3GPP), TS 38.104. [Online]. Available: http://www.3gpp.org/DynaReport/38104.htm
[53] "NR; Physical channels and modulation," 3rd Generation Partnership Project (3GPP), TS 38.211. [Online]. Available: http://www.3gpp.org/DynaReport/38211.htm
[54] "NR; Multiplexing and channel coding," 3rd Generation Partnership Project (3GPP), TS 38.212. [Online]. Available: http://www.3gpp.org/DynaReport/38212.htm
[55] "NR; Radio Link Control (RLC) protocol specification," 3rd Generation Partnership Project (3GPP), TS 38.322. [Online]. Available: http://www.3gpp.org/DynaReport/38322.htm
[56] "NR; Packet Data Convergence Protocol (PDCP) specification," 3rd Generation Partnership Project (3GPP), TS 38.323. [Online]. Available: http://www.3gpp.org/DynaReport/38323.htm
[57] "Evolved Universal Terrestrial Radio Access (E-UTRA) and NR; Service Data Adaptation Protocol (SDAP) specification," 3rd Generation Partnership Project (3GPP), TS 37.324. [Online]. Available: http://www.3gpp.org/DynaReport/37324.htm
[58] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," Internet Engineering Task Force (IETF), RFC RFC 8200.
[59] "Study on NR industrial Internet of Things (IoT)," 3rd Generation Partnership Project (3GPP), TR 38.825. [Online]. Available: http://www.3gpp.org/DynaReport/38825.htm
[60] P. Popovski *et al.*, "Wireless access in ultra-reliable low-latency communication (URLLC)," *IEEE Trans. on Commun.*, vol. 67, no. 8, pp. 5783–5801, 2019.
[61] "Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project (3GPP), TR 38.901. [Online]. Available: http://www.3gpp.org/DynaReport/38901.htm
[62] J. R. Perez *et al.*, "Empirical characterization of the indoor radio channel for array antenna systems in the 3 to 4 ghz frequency band," *IEEE Access*, vol. 7, pp. 94 725–94 736, 2019.
[63] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv*, no. 1412.3555, 2014.



Benoît-Marie ROBAGLIA enters ENSAE Paris in 2014 to study Statistics and Economics before specializing in Machine Learning at the Master Data Sciences at Ecole Polytechnique. After a first experience in the quantitative research lab of BNP Paribas, he joined a newly created consulting start up dedicated to bring AI solutions to the corporate world for one and a half year. He joined Telecom Paris in 2020 as a PhD student in order to work on Deep Reinforcement Learning for uncoordinated Multiple Access.

Marceau COUPECHOUX received the engineering degree of Telecom Paris in 1999, the engineering degree of University of Stuttgart in 2000, the Ph.D. degree from Institut Eurecom in 2004, the Habilitation degree from Sorbonne University in 2015. He is a Professor at Telecom Paris and a Professeur Chargé de Cours at Ecole Polytechnique, Institut Polytechnique de Paris. From 2000 to 2005, he was with Alcatel-Lucent in Bell Labs and then in the Network Design Department. He was a Visiting Scientist with the Indian Institute of Science, Bengaluru, India, in 2011–2012. He has been a General Co-Chair of WiOpt 2017 and Gamenets 2019. In the Computer and Network Science Department of Telecom Paris, he is working on cellular networks, wireless networks, ad hoc networks, cognitive networks, the internet of things, focusing mainly on performance evaluation, optimization and resource management.



Dimitrios TSILIMANTOS received the Diploma and Ph.D. degree in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece, in 2004 and 2009 respectively. From 2011 to 2014 he was with the National Research Institute in Informatics and Control (INRIA) in Lyon, France, as a postdoctoral researcher. Since 2014 he has been with Huawei Technologies, Paris Research Center, France, where he is currently a principal researcher in the Advanced Wireless Technology Lab. His research interests include machine learning, multi-agent settings, network planning, resource management, energy efficiency, stochastic geometry, decision theory and video streaming, with main focus on cellular and wireless networks.