Learning to Solve Decision Problems over two Timescales: An Application to 5G Puncturing

Martin Randall[1*], Gonzalo Belcredi[1], Pablo Belzarena[1] and Federico Larroca[1]

[1]Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de la República, J. Herrera y Reissig 565, Montevideo, 11200, Uruguay.

*Corresponding author(s). E-mail(s): mrandall@fing.edu.uy;
Contributing authors: gbelcredi@fing.edu.uy; belza@fing.edu.uy; flarroca@fing.edu.uy;

Abstract

One of the biggest innovations on 5G and beyond is the support of three different services with particular delay and bandwidth requirements, such as Massive Machine Type Communications (MMTC), enhanced Mobile Broad Band (eMBB) and Ultra-Reliable Low Latency Communications (URLLC). In order to achieve these multiple service requirements, all users have to share resources over the 5G Orthogonal Frequency-Division Multiple Access (OFDMA) frame. One of the strategies proposed by the 5G standard is puncturing, which allows the scheduler to assign eMBB services on a timescale, and on a shorter timescale to preemptively overwrite part of the eMBB assignment when a URLLC user arrives. The optimization of puncturing poses a challenging problem: the optimal allocation depends on traffic arriving over different timescales, which forces the scheduler to make allocation decisions without knowledge of future users' demands, all while having to satisfy several strong constraints. This kind of multiple timescales optimization with restrictions is also to be found in many interesting problems, such as energy management. We propose a learning mechanism where the system learns offline the optimal allocation according to the network state. This learned estimation is then used online to determine the optimal allocation. Through simulations, we verify that the proposed learning strategy provides results close to the optimal policy, improving state of the art proposals for puncturing schemes.

Keywords: 5G and beyond, resource allocation, puncturing, machine learning, eMBB, URLLC

1 Introduction

Next generation wireless networks introduce many interesting challenges, enabling and promoting the use of machine learning in order to optimize resources and services [1, 2]. From beam selection in massive Multiple Input Multiple Output schemes [3, 4] to Non-Orthogonal Multiple Access (NOMA) [5], from cyber security [6] to user centered QoE [7, 8], the machine learning revolution has set foot on open problems arising from new generation wireless communications. We focus here in the coexistence and resource sharing of two different types of users in 5G and beyond: the classical high bandwidth user (eMBB - enhanced Mobile Broadband) and the innovative reliable and low latency new users (URLLC - Ultra-Reliable Low Latency Communications). Each of these services has very different characteristics and requirements. While eMBB users demand high data rates and thus an important set of resources of the OFDMA (Orthogonal Frequency Division Modulation Access) grid, URLLC services require high reliability and very low latency [9].

In order to satisfy the aforementioned requirements, the 5G New Radio (NR) standard proposes different mechanisms [10], among them the so-called puncturing scheduling. In this case both services are assigned resources on the OFDMA grid at different timescales: slots for eMBB and (shorter) minislots for URLLC (see Figure 1). The main idea behind puncturing is to assign with preemptive priority URLLC traffic over eMBB resources. In this case, the scheduler assigns resources to each eMBB user every time slot, but URLLC traffic are assigned on a minislot timescale.

The resulting resource allocation problem is inherently complex given the coupled nature of the two timescales involved (URLLC and eMBB scheduling), and the uncertainty of future URLLC demand: the scheduler needs to assign resources to eMBB users so as to maximize a chosen metric (i.e. throughput for eMBB users), resources that may be later overwritten in order to satisfy URLLC requirements, unknown at the moment eMBB resource are scheduled. Naturally, assigning an OFDMA resource during a minislot to a URLLC user on a resource already assigned to an eMBB user implies that the latter will receive, at least, part of the corresponding frame with errors. Thus the scheduler has to solve two optimization problems at different timescales but highly coupled. Adding further difficulty, the scheduling decisions have to be made in a very narrow time lapse.

Naturally, this resource allocation problem has attracted much research attention [11]. For instance, the performance of specific assignment policies have been extensively studied [12–15]. On the other hand, some works formulate the optimization problem on puncturing in 5G NR and search for

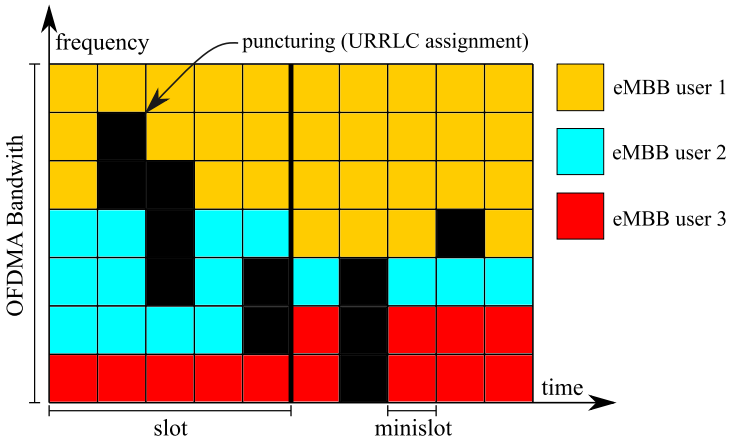

Fig. 1 Puncturing scheme. The OFDMA grid is split in time (slots and minislots) and frequency. While resources to eMBB users are assigned at the slot timescale, URRLC users' resources are allocated at the minislot timescale, overwriting eMBB data and possibly resulting in errors for the corresponding eMBB receiver.

heuristics that behave reasonably well for particular given scenarios [16, 17]. Other authors propose a Q-learning or deep-learning approach to obtain a solution to the puncturing problem [17, 18]. A very recent and interesting article proposes the utilization of reconfigurable intelligent surface (RIS) in order to optimize eMBB and URLLC coexistence [19]. These works allow us to visualize how puncturing behaves in certain scenarios and provide insight into the problem. We particularly remark the paper [20], where authors formalize the puncturing optimization problem and analyze some practical interesting cases. An interesting observation in [20] is that when the problem's functions (loss, constraints) are non-convex (e.g. threshold loss) and as URLLC's traffic load grows their proposed methods for puncturing increasingly differ from the optimal solution. It is important to note that, as discussed for instance in [21], bit error rate (BER) on OFDM systems follows a threshold-like phenomena with noise and interference. As puncturing increases, such behaviour is expected, highlighting the relevance of finding policies that are well-adjusted to threshold losses.

It is important to highlight that this kind of highly coupled optimization problems with multiple time basis and hard constrains can be found in many fields besides wireless communications. For instance, a two timescales problem arising from long term storage management is studied in [22]. Similar problems are studied in the more general field of resource allocation [23], as in [24] where a two timescales coupled optimization problem arises from management of IoT devices, or [25] where authors use supervised learning in order to solve wireless IoT networks' resource allocation problems. The authors of [26] study a two timescales optimization problem arising from micro grid scheduling, and [27] studies the optimization problem given by the need to make predictions over energy consumption for electrical vehicles charging on two timescales.

In order to decide now the best resource allocation without knowing future puncturing needs, we propose a supervised learning framework that learns to approximate both scheduling policies. The key observation is that, although the optimal solution for our coupled optimization problem is unavailable at the scheduling intervals, it can be found afterwards, once events have happened and we can consider all necessary data from a certain time slot (and its corresponding minislots). If the loss due to puncturing penalizes the eMBB user's utility through a convex loss function, the problem turns out to be a convex optimization problem, which has led to numerous and well known solutions [20]. But if the loss penalty is not convex (for example, a threshold penalty, a much more suitable model for the actual wireless communication link [21]), as we mentioned before proposed solutions can grow afar from optimal. In such cases, the optimal policy would have to be found by trying all possible scheduling scenarios, making it impossible to find online.

Following the ideas of the 'learning to optimize' paradigm (as in [28] and [29]), we propose a statistical learning method that enables the exploitation of signal's correlations in order to approximate optimal solutions. The proposed learning framework separates the resource assignation problem over each timescale, although incorporating significant statistical information. This is achieved by training two learning machines (agents) that will be decision makers for each timescale resource assignation. As we show in the simulations section, this method has shown to be well suited for learning when faced with non-convex loss functions, allowing the proposed learning formulation to surpass current state of the art policies for the eMBB and URLLC coexistence dilemma. Crucially, since actual resources are assigned in the inference phase of the learning algorithms (i.e. costly training is peformed offline) the computational cost of the proposed method's operation is extremely lightweight.

All in all, the main contributions of this work are the general formulation we propose to deal with a two timescales coupled optimization problem, following the learning to optimize paradigm. We also provide a concrete application to the 5G NR problem of scheduling eMBB and URLLC users through puncturing. Our adaptive learning framework allows us to find a close to optimal policy given any loss function. This distinguishes our practical approach from similar state-of-the-art proposals, enabling more realistic scenarios. As succinctly presented in Table 1, when compared to other methods available in the literature, our proposal is able to closely approximate optimal policies all while being computationally amenable to an online operation.

In Section 2 we define a mathematical formulation for the puncturing problem and detail the proposed learning framework. This general problem, which as we mentioned before appears in several other domains, is instantiated in the URLLC and eMBB puncturing case in Section 3, using it as a use case example and proposing different system reward functions, formulating the offline optimization problems with which we get the optimal solutions. As we present in Section 4 our proposal yields state of the art results for convex loss functions,

Table 1 Comparative analysis of State-of-the-Art proposals.

Proposal	Minislot scheduling	Puncturing lossess			
		Linear	Quadratic	Threshold	Other
[13–15](Round Robin heuristic)	✗	✓	✗	✗	✗
[16](House Allocation Policy heuristic)	✓	✓	✗	✗	✗
[17](Mixed integer linear programming-deep reinforcement learning)	✓	✓	✗	✗	✗
[18] (Q-learning)	✗	✓	✗	✗	✗
[19]a (Alternating optimization based Heuristic)	✗	✓	✓	✓	✗
[19]b (Heuristic)	✗	✓	✓	✓	✗
[20] (Gradient scheduler based optimization)	✓	✓	✓[2]	✓[2]	✗
Ours: Learning to Optimize paradigm	✓	✓	✓	✓	✓

[2] For certain scenarios and specific URLLC traffic distributions they find approximately optimal policies.

while allowing for a much closer approximation to the optimal solution than most widely used algorithms in the non-convex case. Furthermore, we compare the results of using several supervised learning algorithms, illustrating that the mathematical formulation of the learning problem is robust and generalizes well with different techniques. This allows for a combination of total or partial handcrafted algorithms (which are widely used in practical applications) with state of the art learning techniques (which are widely used in scientific research).

2 System Model and Problem Statement

Let us consider the following optimization problem. A finite and fixed amount of a certain resource has to be distributed over time to different users, who will in turn profit from the assignation by obtaining an utility. Consider two groups of users: E primary users, and U secondary users that will be puncturing primary users. In a more general formulation, instead of users there could be simply two sets of resource consumers. The second group of resource recipients has strong requirements with regard to the resource utility to be satisfied (a certain demand to fulfill), which implies the utilization of resources already assigned to the first set. A decision maker has to make resource assignations to both sets, satisfying the second group's demands all the while trying not to harm the first group's utility.

We will consider that time is slotted on two scales: a generic time slot t of length T will contain M minislots indexed by τ. At each time slot t there is a resource assignation decision to make, represented by the vector $\mathbf{x}[t] \in \mathbb{R}^E$ assigning resources to each of the E primary users during time slot t. The vector $\mathbf{x}[t]$ can be constructed, for example, by indicating in its e-th entry the amount of resources assigned to user e. Similarly, for every minislot τ of time slot t, there is a resource assignation matrix $\mathbf{y}[t, \tau] \in \mathbb{R}^{U \times E}$. In the same way as with $\mathbf{x}[t]$, this second resource assignation variable indicates on its u, e entry the amount of resources of each primary user e reassigned to the secondary user u. By summing over row u of matrix $\mathbf{y}[t, \tau]$ we get the total number of resources assigned to user u, while by summing over column e we get the total number of resources previously assigned to e and now reassigned to users of the U group. We will denote as $\mathbf{y}[t] = [\mathbf{y}[t, 0], \mathbf{y}[t, 1] \dots \mathbf{y}[t, M-1]] \in \mathbb{R}^{E \times U \times M}$ the minislot based assignations during time slot t.

We will consider two kinds of random variables. On the one hand a random vector $\mathbf{R}[t] \in \mathbb{R}^{E+U}$, that will be drawn at the beginning of each time slot. This vector represents the user's ability of exploiting the resources; for instance, maximum attainable rate over a wireless channel for all users, or an electrical generation capacity. On the other hand, random vector $\mathbf{D}[t] = [\mathbf{D}[t, 0], \mathbf{D}[t, 1] \dots \mathbf{D}[t, M-1]] \in \mathbb{R}^{M \times U}$, with realizations for every minislot τ in time slot t. This vector represents a certain requirement for secondary users; for example, a data rate demand to be satisfied, or an electrical energy requirement. As we mentioned, these requirements only correspond to secondary users, and we consider primary users to be greedy towards resources (resources are shared among them in order to maximize some utility). It is important to note that random vector $\mathbf{D}$ has realizations on a minislot basis, thus not being known when resource assignation to primary users happens. Suppose all random vectors ($\mathbf{D}$ and $\mathbf{R}$) and policy assignments ($\mathbf{x}$ and $\mathbf{y}$) are coupled through a set of H restrictions. Choosing both assignment policies in order to maximize a certain function of the variable leads to a very general formulation as a constrained stochastic optimization problem:

$$\max_{\mathbf{x}[t], \mathbf{y}[t]} f(\mathbf{x}[t], \mathbf{y}[t], \mathbf{R}[t], \mathbf{D}[t]) \tag{1}$$

s.t.:

$$g_i(\mathbf{x}[t], \mathbf{y}[t], \mathbf{R}[t], \mathbf{D}[t]) \leq 0, i = 0, \dots, H-1. \tag{2}$$

These optimization problems depending on random variables with realizations posterior to the decision instant are very hard to solve. Note the difficulty of making resource assignation decisions at the beginning of a time slot when resource demands and resource assignation decisions for the same time period are unknown and bound to affect all utilities. Our proposed supervised learning framework on its most general form will try to approximate the optimal

solution $(\mathbf{x}^*, \mathbf{y}^*)$ in the following manner. After a time slot has elapsed, all random variables realizations are known and the optimal solution can be found offline. Therefore the approximate solution will be learned from the combined data set of the random variables' realizations and the optimal assignation policy. The set of constrains has to be ensured after the supervised agents make their assignations, which can be achieved through some projection over the feasible space. The algorithm consists of the following steps during training:

1. Given K previous realizations of the random variables $\mathbf{R}$ and $\mathbf{D}$, we solve the optimization problem and find optimal assignments $(\mathbf{x}^*[t], \mathbf{y}^*[t])$ for every $t \in [0, K]$.
2. We train supervised learning agent 1 with available random variables realizations as features and the optimal assignments as targets for every t. The features may include the last K_1 realizations. As an example, for time slot q, realizations $\mathbf{R}[t]$ with $t \in [q - K_1 \dots q]$ and $\mathbf{D}[t]$ with $t \in [q - K_1 \dots q - 1]$ will be considered as features and $\mathbf{x}^*[q]$ will be choosen as target.
3. We train a second supervised learning agent on a minislot basis, considering the last $K_2 \times M + p$ minislots' realizations of random vectors $\mathbf{D}$ and $\mathbf{R}$ in order to predict an approximation to the optimal policy for minislot p in time slot q. In this case, features will be $\mathbf{R}[t]$ with $t \in [q - K_2 \dots q]$, $\mathbf{D}[t]$ with $t \in [q - K_2 \dots q - 1]$, and $\mathbf{D}[q, \tau]$ with $\tau \in [0 \dots p]$. Finally, another feature that will be used is the found value of $\mathbf{x}^*[q]$. Our target will be the minislot assignation policy $\mathbf{y}^*[q, p]$.

Together, our learned agents will be able to forecast online resource assignations $(\hat{\mathbf{x}}^*[t], \hat{\mathbf{y}}^*[t, \tau])$. Note that the training phase is done offline, with no time constrains, which enables step 1 to eventually solve non-convex optimization problems. Also observe that when using a time slot basis, mini slot features have to be somehow summarised into a time slot basis (for example, as the aggregated demand over the time slot); similarly, time slot based data has to be transformed into mini slot based realizations. Both learning agents offer the freedom to choose architectures over the vast world of supervised learning, according to the problem's nature and/or domain knowledge over the problem and random variables $\mathbf{R}$ and $\mathbf{D}$.

When working online, the procedure is very simple:

1. At time slot t, features are available for supervised agent 1 to predict assignation $\hat{\mathbf{x}}[t]$.
2. Projection of $\hat{\mathbf{x}}[t]$ if needed in order to satisfy restrictions over $\mathbf{x}$.
3. For every minislot τ, features are available for the second agent to predict assignation $\hat{\mathbf{y}}[t, \tau]$.
4. Projection of $\hat{\mathbf{y}}[t, \tau]$ if needed in order to satisfy restrictions over $\mathbf{y}$.

It is important to remark that the projection is needed in order to satisfy restrictions and propose policies belonging to the feasible space of solutions. There are different ways of projecting that can be used in order to ensure that restrictions are complied, which we will further discuss in subsection 3.3.

In the next section we introduce a practical example of the frameworks' utilization, through the coupled problem that arises from URLLC and eMBB coexistence in 5G NR.

3 URLLC and eMBB coexistence in 5G NR

As we have introduced, 5G NR establishes three types of users. Two of these users, URLLC and eMBB, share resources of the OFDMA grid. But their coexistence comes at a price: puncturing over eMBB assigned resource is done in order to satisfy URLLC demand. This degrades performance for eMBB users, which poses a challenge: to minimize puncturing effects on eMBB users, all the while satisfying URLLC demand. Because URLLC's demand and allocation happens at more frequent intervals, eMBB resource assignation and puncturing which may be studied as a coupled optimization problem over two timescales, as those discussed in section 1.

3.1 Definitions and optimization problem

In the frequency domain, the OFDMA grid is divided into sub-channels, whereas in the time domain it is split into slots and minislots (a set of OFDM symbol times). The 5G NR standard defines the minimum set of OFDMA resources that can be assigned to an eMBB user: a Resource Block -RB-, which in this paper corresponds to what we call a time slot and a set of sub-channels. Considering the complete OFDMA grid, the system disposes of a total of N_{RB} resource blocks.

At each slot t a first scheduler distributes among active eMBB users the set of resource blocks. The system has a set of eMBB users in each time slot t: $e \in \{1, .., E\}$. We will assume that eMBB users' traffic follows a full buffer model. Figure 2 pictures actions from the first scheduler with respect to eMBB users.

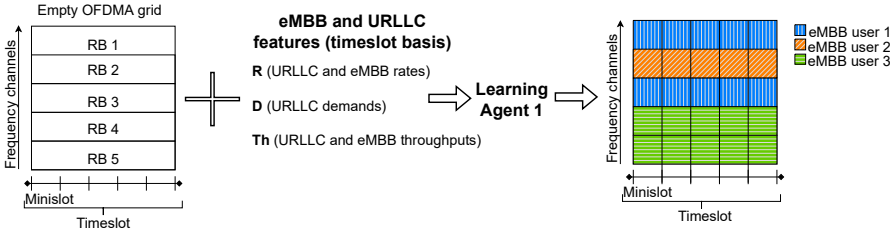

Fig. 2 First resource allocation scheduling, corresponding to eMBB users on a time slot basis. Scheduling of eMBB users is prior to scheduling of URLLC users. Features used by the scheduling agent are composed of eMBB and URLLC rates, throughputs and demands of past time slots.

As seen in the previous section, and using the same notation for ease of exposition, we define a time slot t as divided into time minislots indexed by

$\tau \in \{1, ..., M\}$. The system has a set of U URLLC users, $u \in \{1, .., U\}$. Each URLLC user u at (t, τ) has a byte demand $d_u(t, \tau)$, which is the realization of a random variable D_u, known at the beginning of each minislot. In order to fulfill URLLC traffic demands, a second scheduler assigns at each minislot resource blocks that share the same set of sub-channels of an RB assigned to an eMBB user, but the time duration of a minislot. Figure 3 allows readers to visualize the minislot based puncturing action for URLLC users, overwriting assigned resources to eMBB users.

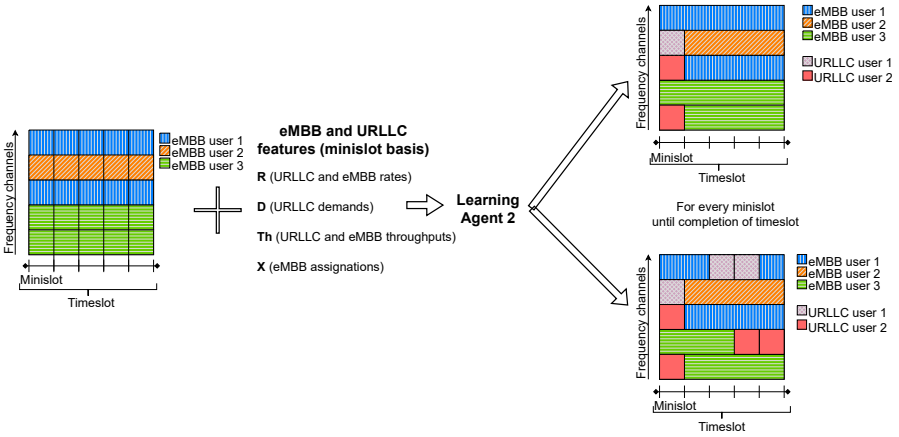

Fig. 3 The second agent schedules resources to URLLC users on a minislot basis. These resources were already assigned to eMBB users, and may thus inflict losses to eMBB communications.

All users have a wireless channel with the base station that varies between time slots depending on the channel's fading and noise. We assume the base station knows the SNR of each user at all times, which is constant over a timeslot. Using the 5G NR standard, the system can find the maximum reachable rate for any user by using the measured SNR and the corresponding coding and modulation. This SNR evolution and its impact on the maximum attainable rate introduces a time slot and user dependent random variable R_e for eMBB users and R_u for URLLC users, with realizations $r_e(t)$ and $r_u(t)$ known at the beginning of each time slot. We consider all users' peak rates at time slot t on vector $\mathbf{R}[t] = [r_e(t), r_u(t)]^{e \in \{0...E\}, u \in \{0...U\}}$ and demands for URLLC users on vector $\mathbf{D}[t] = [d_u(t)]^{u \in \{0...U\}}$.

Typically these random vectors have correlations between them and temporal correlations during consecutive time slots and minislots. These dependencies are to be exploited in order to learn accurate predictions, as well as the impact of the scheduling policies on the overall utility.

We consider the vector $\mathbf{x}[t] = \{x_1(t), ..., x_E(t)\}$, where $x_e(t)$ represents the total number of RBs assigned in slot t to eMBB user e (e.g. $\mathbf{x}[t] = [2, 1, 2]$ in figure 2). We define as well the RB assignation to URLLC users as the vector

$\mathbf{y}[t] = \{y_1(t), ..., y_E(t)\}$, where $y_e(t)$ is the sum over all minislots on slot t of the number of punctured RBs being assigned to eMBB user e (e.g. $\mathbf{y}[t] = [0.6, 0.2, 0.6]$ in figure 3). In a similar way, we represent by $y_{e,u}(t, \tau)$ the number of RBs reassigned from eMBB user e to URLLC user u at time slot t and minislot τ (e.g. $\mathbf{y}_{2,1}(t, 0) = 0.2$ in figure 3), and thus $y_u(t, \tau) = \sum_e y_{e,u}(t, \tau)$ is the total resource block assignation for URLLC user u.

Please note that for a given URLLC user u, its byte demand $d_u(t)$ has to be satisfied over a minislot, meaning that assignation $y_u(t, \tau)$ multiplied by the time slot peak rate $r_u(t)$ has to equal the users demand. As puncturing of a certain eMBB user cannot exceed its original total amount of resources, $y_e(t) \leq M x_e(t)$, $\forall e, t$. Combining the aforementioned analysis, puncturing has to verify the following equations:

$$y_e(t) = \sum_{\tau=1}^{M} \sum_{u=1}^{U} y_{e,u}(t, \tau) \tag{3}$$

$$d_u(t, \tau) = \sum_{e=1}^{E} y_{e,u}(t, \tau) \times r_u(t) \tag{4}$$

As we mentioned before, we want the resource allocation to eMBB and URLLC users to be performed so that a certain utility function f is maximized. We will consider this reward to be a function of random variables realizations: $\mathbf{r}[t]$, $\mathbf{d}[t]$ and of course of assignations $\mathbf{x}[t]$, $\mathbf{y}[t]$. In particular, the decision on $\mathbf{y}[t]$ will impact on the performance as perceived by the eMBB users, as part of their transmission is overwritten.

We assume that by knowing a time slot's realizations (and past if needed), the scheduler can find the optimal assignment in the present slot t:

$$\mathbf{x}^*[t], \mathbf{y}^*[t] = \arg \max_{\mathbf{x}[t], \mathbf{y}[t]} f(\mathbf{x}[t], \mathbf{y}[t], \mathbf{r}[t], \mathbf{d}[t]) \tag{5}$$

The total utility of our system f will be defined as the sum over eMBB users' utility, which is in turn defined as a per user utility function $\mathcal{U}$ constructed as some combination over eMBB users' throughput. As an example, we could use as global and per user utilities $f = \sum_e \mathcal{U}(th_e(t)) = \sum_e \log(1 + th_e(t))$. Throughput $th_e(t)$ for user e at slot t is defined as a running mean with averaging ratio α over the effective rate (7), which is defined as a function of the assigned resources $x_e(t)$, maximum attainable rate $r_e(t)$ and, as we mentioned before, the rate loss due to puncturing of user's e transmission (a function we will denote as $L(x_e(t), y_e(t))$). We consider throughput for all eMBB users at time t in vector $\mathbf{th}[t] \in \mathbb{R}^E$.

$$f(\mathbf{x}[t], \mathbf{y}[t], \mathbf{r}[t], \mathbf{d}[t]) = \sum_e \mathcal{U}(th_e(t)) \tag{6}$$

$$th_e(t) = \alpha th_e(t-1) + (1-\alpha) c_e(t) \tag{7}$$

$$c_e(t) = x_e(t) r_e(t) (1 - L(x_e(t), y_e(t))) \tag{8}$$

All in all, the resulting optimization problem is as follows:

$$\max_{\mathbf{x}[t], \mathbf{y}[t]} \sum_{e=1}^{E} \mathcal{U}(th_e(t)) \tag{9}$$

s.t.:

$$th_e(t) = \alpha th_e(t-1) + (1-\alpha) c_e(t) \tag{10}$$

$$c_e(t) = r_e(t)(1 - L\left(x_e(t), y_e(t)\right)) \tag{11}$$

$$\sum_e x_e(t) = N_{RB} \tag{12}$$

$$\sum_u y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ t, e, \tau \tag{13}$$

$$y_e(t) = \sum_\tau \sum_u y_{e,u}(t,\tau) \ \forall \ t, e \tag{14}$$

$$\sum_e y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{r_u(t)} \ \forall t, \tau, u \tag{15}$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall \ e, u, t, \tau \tag{16}$$

$$x_e(t) \geq 0 \ \forall \ t, e \tag{17}$$

Restrictions ensure that definitions hold to the puncturing model. For instance, any resource assignation has to be positive, and there cannot be more resources punctured to an eMBB user than the ones originally assigned. Constraints will be further analyzed in subsection 3.3.

Let us further discuss the loss function L. As puncturing of eMBB and URLLC in 5G is a hard to solve optimization problem, different scenarios have been proposed in the literature, mainly using heuristics yielding good results for particular settings. Other approaches, as [20], formulate optimization problems very similar to our own, and consider convex functions for $\mathcal{U}$ and L. They prove that some policies are optimal under certain conditions, notably a proportional fair assignation for eMBB resources and a random assignation for URLLC resources. However, digital communications' performance exhibit a non-convex response to interference, that can be viewed as a threshold penalty: if interference is larger than a certain value, then the message is completely lost [21].

We will thus test our framework using two kinds of loss functions: convex (quadratic) and the more realistic non-convex threshold. This allows a comparison with regard to optimal solutions with theoretical guarantees (as in the convex scenarios), as well as with state of the art proposed solutions on the threshold case, which is a much more realistic approach, and closer to application. Our proposed learning method proves to be able to generalize to different scenarios all while maintaining close to optimal mean utility for eMBB users.

The offline optimization phase may be performed using any chosen solver when the loss and utility functions are convex. For the threshold scenario, even if this is not a convex optimization problem, it can be transformed and solved by optimizing 2^E convex optimization problems. Even though challenging, the computational complexity of solving the threshold scenario only depends then on the number E of EMBB users, and neither on the number of active URLLC users nor the number of resource blocks the system disposes.

As stated, in a general non-convex scenario, the optimal policy has to be found by trying all possible policies, which is highly time and resources consuming. Still, in our framework this is feasible since solving the optimization problem and the training phase are both done offline.

3.2 Learning formulations

An important consideration to be made over resource block assignations and puncturing is the strong time limitation a 5G scheduler meets. As an example to fix this idea, a time minislot in 5G is on the order of the hundreds of nanoseconds. This is why supervised learning is an appropriate choice in order to exploit statistical information: once trained, execution is very quick.

Different supervised learning methods could be exploited. The chosen learning machine will typically depend on domain knowledge, mainly on the nature of the random variables, depending if and how much we know about them. The range of possible choices goes from well known methods as Random Forest or Support Vector Machines up to the newest deep learning architectures, including custom made algorithms. In section 4 we show results with different learning agents.

Whichever learning method is chosen, with our proposed statistical learning framework they will all share features and target. Our first scheduler makes the prediction of $\mathbf{x}^*[t]$, for which we use the following features:

$$feats_{\mathbf{x}^*[t]} = [\mathbf{r}[t] \dots \mathbf{r}[t - K_R], \mathbf{d}[t-1] \dots \mathbf{d}[t - K_D], \mathbf{th}[t-1] \dots \mathbf{th}[t - K_{th}]] \quad (18)$$

We include rate for all users (of both kind) for this time slot and past K_R time slots. We also consider eMBB users' throughput and URLLC demand for the past K_D and K_{th} time slots respectively; observe that neither eMBB's throughputs nor URLLC's demands are available at present time slot t. Demand from URLLC users has a different timescale, so including the minislot demand has little significance. Instead, we consider the aggregated sum of URLLC users' demands over each time slot. With regard to the more general formulation described in section 2, note that we have introduced as a feature the throughput instead of the assignation x, both being directly related: with the assignations and rates the throughput can be easily obtained. We consider the throughput to be a more direct feature than to introduce the past assignations for the present learning poblem.

Our second scheduler will be working on a minislot time basis. In order to match dimensions over a minislot basis, timeslot features are extended to the M minislots (e.g. rates). Besides past and present rates for all users and throughput for eMBB users, we use URLLC users demands per minislot (past and present), and resource assignation for eMBB users $\mathbf{x}^*[t]$. This results in:

$$feats_{\mathbf{y}^*[t,\tau]} = [\mathbf{r}[t], \mathbf{d}[t,\tau] \dots \mathbf{d}[t,\tau - K_D], \mathbf{th}[t-1], \mathbf{x}^*[t]] \tag{19}$$

3.3 Optimization constraints

The regression problem formulated in our approach must take into account the constraints of the system. In the estimation of the eMBB assignments, we must ensure the following constraints:

$$\sum_e x_e(t) = N_{RB} \tag{20}$$

$$x_e(t) \geq 0 \ \forall \ e \tag{21}$$

Regarding the estimation of the URLLC allocation we must ensure the following constraints:

$$\sum_u y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ e,t,\tau \tag{22}$$

$$\sum_e y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{r_u(t,\tau)} \ \forall \tau, u, t \tag{23}$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall \ e,u,t,\tau \tag{24}$$

These constraints must be imposed on the learning regression system. There are two types of constraints in the previous equations. The first type guarantees that the output vector of the regression ($\mathbf{x}[t]$ or $\mathbf{y}[t,\tau]$) has all terms greater or equal than zero and the sum of its components must equal some fixed value: the total amount of resources for $\mathbf{x}$, and the demand satisfaction for $\mathbf{y}$. This type of constraints can be imposed using a well known method consisting of a logarithmic transformation over the normalized target vector, as is described in [30]. This transformation ensures that the regression output satisfies fixed sum constraints.

The other type of constraint is:

$$\sum_u y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ e,\tau \tag{25}$$

This constraint must be imposed in the regression of the $\mathbf{y}$ values where the x_e values were calculated in the first regression. This type of constraint

is more difficult to impose together with the first type of constraints. In this case after the regression with the log transformation, we assure that all the constraints for $\mathbf{y}$ are verified by projecting the output values obtained $\mathbf{y}^*[t,\tau]$ to the constraint space.

$$\min_{\mathbf{y}} \sum_{e} (y_{e,u}(t,\tau) - \hat{y}_{e,u}(t,\tau))^2 \tag{26}$$

s.t.:

$$\sum_{e} y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{\hat{r}_u(t)} \ \forall u \tag{27}$$

$$\sum_{u} y_{e,u}(t,\tau) \leq x_e(t) \ \forall e \tag{28}$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall e,u,t,\tau. \tag{29}$$

The projection is realized for each minislot τ.

4 Simulations and Results

In order to validate our system model and learning procedure, we compare our results to the proposed solution of [20], which we will refer to as 'baseline' in experiments and tables. In a nutshell, this heuristic consists of a proportional fair assignation for eMBB users and a random assignation for URLLC puncturing. This comparison might be unfair, given that the solution is only optimal in certain specific scenarios (for example, for convex loss functions), but it is a state of the art algorithm for puncturing optimization, and as such serves well as a baseline in order to compare our proposed learning procedure behaviour. We use well known supervised learning methods such as the classic Support Vector Machines (SVM) and different flavours of the more modern Recurrent Neural Network (RNN), and implement the learning algorithm over different synthetic scenarios.[1]

We selected three particularly interesting scenarios. On the first experiment we apply our proposed framework to a convex problem (both loss function and utility are convex). This type of 'convex' scenarios have well known solutions (as shown in [20]), which enable us to test a simple setting in order to verify the correct behaviour of our learning agents. On the second and third scenarios we use a threshold loss: if more than a certain amount of assigned resources are punctured for an eMBB user, all communication will be considered lost (meaning the rate for that user will be 0 for the whole time slot). On the other hand, if that threshold is not surpassed, communications (and thus eMBB user rates) are unaltered. This non-convex scenarios are studied by [20], and while their proposed heuristic works reasonably well when URLLC demand is low, as traffic increases the proposed solution grow afar from optimal policies. That is

[1] All code can be found at https://gitlab.fing.edu.uy/ai45g/puncturing-ai45g.

why we finally introduce a synthetic scenario with larger URLLC traffic loads, maintaining the threshold penalty as the loss function. As mentioned before, the system's reward is defined as $f(t) = \sum_e \mathcal{U}(th_e(t)) = \sum_e \log(1 + th_e(t))$.

Besides our proposal and the solution described in [20], we compare our results with a 'random agent'. This agent will assign a fixed number of resource blocks to eMBB users (total number of resource blocks over number of eMBB users), and a random number of resource blocks to URLLC users. The idea with this extremely simple agent is to obtain a minimum performance so as to quantify the gain obtained by the other two methods. For all heuristics we ensure restrictions are met by projecting the found resource assignations over the feasible space of solutions.

Simulation parameters can be found in table 2. The total number of resource blocks is $N_{RB} = 270$ for all simulations. Channel evolution for all users follows a finite Markov process as proposed in [31]. The URLLC users' traffic demand follows a two state finite Markov process, being URLLC's demand turned on and off. All convex optimization problems were solved using **cvxpy** [32].

When using SVM, the regression for learning both policies is obtained by using Scikit-learn Support Vector Regression (SVR) software[33]. Parameters C and gamma of the Radial Basis Function (RBF) kernel are obtained via grid search.

Recurrent Neural Networks are well known deep learning architectures particularly suited to learn and capture sequential dependencies (e.g. temporal sequences), and are widely used in natural language processing. These properties make them a natural choice for our proposed statistical learning method. We used the keras [34] implementation of recurrent neural networks, experimenting with LSTM, GRU and vanilla RNN. We only present experiments concerning the implementation achieving highest results, and we used a very simple architecture: an LSTM (or GRU) layer followed by a fully connected layer for regression purposes. In all cases, the learning rate follows an exponential decay and the loss function is computed by using mean squared error. The parameters of the RNN (number of neurons, learning rate, normalization, sequence length) were also chosen using grid search, and details for each experiment can be found in table 2. When using neural networks, normalization was used prior to learning in order to accelerate convergence and obtain better scores.

We first tried the proposed learning method using a quadratic loss function, defined as $L(x_e, y_e) = (\frac{y_e}{x_e \times M})^2$. The predicted policy obtained is very close to optimal as can be seen in figure 4, and the average utility is over 99% of the optimal utility (see tables 3, 4). Observe that in this case all policies behave really well; in a convex scenario the puncturing policy is not that important in order to achieve good results, and an average distribution of resources over eMBB users will be optimal in expectation. Results are good but also expected, because by using a simple convex loss function the optimal policy should be easily learned. This is caused because losses by puncturing

Table 2 Table of Parameters

	Scenario 1	**Scenario 2**	**Scenario 3**
Loss model	**Quadratic**	**Threshold**	**Threshold**
eMBB users	20	10	10
URLLC users	5	3	3
Minislots (M)	3	3	3
α	0.8	0.8	0.8
Threshold	-	0.11	0.11
T_{train}	7700	7700	19700
T_{test}	300	300	300
SVM-C (x)	$50e^3$	1000	200
SVM-G (x)	$1e^{-4}$	$1e^{-6}$	$5e^{-6}$
SVM-C (y)	100	$5e^3$	$20e^3$
SVM-G (y)	$1e^{-7}$	$5e^{-6}$	$5e^{-7}$
Max Iterations	$1e^4$	$1e^5$	$1e^5$
RNN Cell	GRU	LSTM	LSTM
Hidden Layer (x)	24	24	24
Hidden Layer (y)	128	32	32
Normalization	No Scaler	MinMax	MinMax
Learning rate (x)	$1e^{-3}$	$1e^{-4}$	$1e^{-4}$
Learning rate (y)	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
Sequence length (x)	2	4	4
Sequence length (y)	6	12	12

are not so dependent on the punctured eMBB user, as all eMBB users fare similarly and contribute proportionally to the global utility. Also, on average a mean distribution of resources for eMBB users will be optimal, meaning the random heuristic proposed should (and effectively does) yield good results. Yet, results are encouraging, as our proposed learning method fares as well as heuristics that are optimal on mean (as the Baseline).

We then applied the framework in a more complex scenario, using the threshold loss function formulation (see figure 5). Results are even more encouraging, even if they are not as near optimal as in the quadratic case, the utility achieved is almost 90% of the utility the optimal policy would obtain. The difficulty when using the threshold loss function is that when an eMBB user is punctured it may result in a null rate ($r_e(t) = 0$), which means the discontinuities observable in the figures and an evident larger impact on the utilities for different policies. For an agent to learn these highly discontinuous patterns is not a trivial task, and yet our proposed method is able to grasp statistical correlations and features' relationships in order to choose close to optimal policies.

In the work of [20], authors find greater differences between their proposed heuristics and optimal solutions as the traffic load increases for URLLC users, when using a non-convex loss function (for example a threshold loss). Even if the discontinuities may appear more often (eMBB users with $r_e = 0$), this may actually be beneficial for the learning agent. For instance, it will learn to puncture those eMBB users with $r_e = 0$, since their limit threshold has already

Fig. 4 Reward during online test for Scenario 1. All heuristics fare almost as well as the optimal solution, given the convex setting for the trial: both loss function (quadratic) and utility are convex. In this case the puncturing policy has a lesser effect on global utility, and a mean distribution of resources for eMBB will have a good performance on expectation.

been surpassed. This is evidenced in the third scenario, where URLLC's traffic load is larger (see figure 6).

As expected, utilities are farther apart from optimal than in previous experiments, although our framework fares much better than the rest of the proposed solutions: while the other heuristics attain less than 50% of optimal utility (and almost vanishing at certain time-slots), our proposed learning method achieves over 75% of the optimal utility in all cases. Results are summarized in tables 3 and 4. Table 3 compares achieved utilities for the optimal policy, our proposed algorithm, the baseline and the random policy. In Table 4, the obtained rewards are normalized with respect to the optimal value to derive a performance metric. In the former mean reward depicts the overall utility obtained with each method, whereas in the latter we present the algorithm's performance measured as the ratio between the achieved utility with regard to the optimal utility.

Table 3 Mean reward for compared algorithms over different scenarios. Reward is calculated by applying a fairness utility function to the eMBB users' throughput as in eq. 6.

	Mean Reward				
Loss model	**Baseline**	**Random**	**SVM**	**RNN**	**Optimal**
Quadratic	92.43	92.47	92.43	**92.48**	**92.63**
Threshold (low demand)	45.30	40.13	45.32	**47.62**	**53.94**
Threshold (high demand)	24.21	16.68	**39.41**	38.79	**50.78**

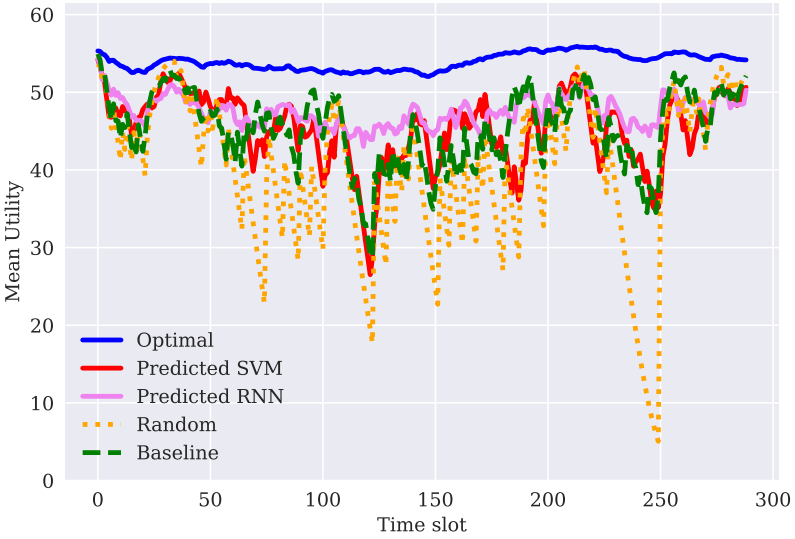

Fig. 5 Reward during online test for Scenario 2 (Threshold Loss). During test, URLLC traffic demand is low, allowing agents to either allocate all puncturing without losses or to puncture one eMBB user over the threshold limit. This scenario is harder to learn, having two very distinct scenarios (either send all puncturing to a user of distribute it evenly among all users).

Table 4 Performance with respect to the optimal policy's utility for proposed and baseline heuristics.

	Performance (%)			
Loss model	**Baseline**	**Random**	**SVM**	**RNN**
Quadratic	99.78	99.83	99.78	**99.84**
Threshold (low demand)	83.98	74.40	84.02	**88.28**
Threshold (high demand)	47.68	32.85	**77.61**	76.39

5 Conclusions

We have presented a framework to solve online two timescales constrained optimization problems. In our proposed learning approach, we use the system's state and statistical correlations in order to train two supervised agents, one for each timescale policy. Our framework consists of an offline learning phase, in which we train the learning agents with the system state and the optimal assignments obtained offline. The supervised agents then apply online the estimated optimal assignments.

The proposed framework is instantiated on resource allocation with puncturing in 5G networks. The optimization of 5G resources with puncturing is

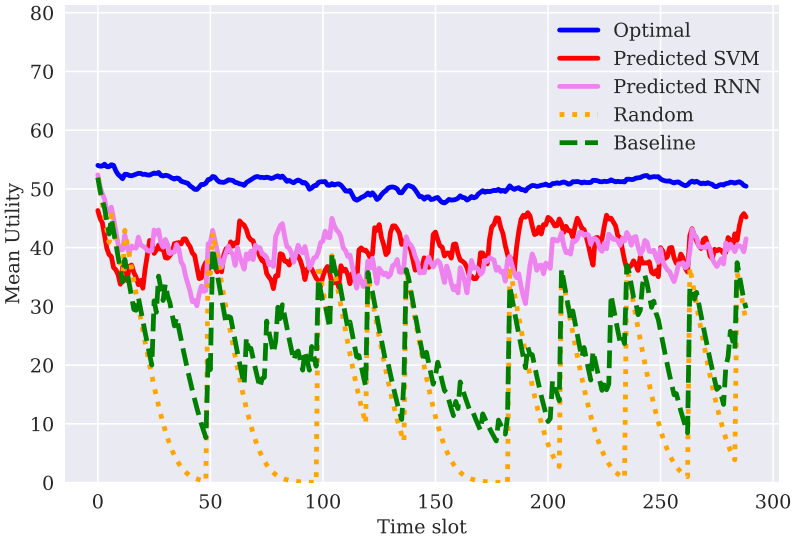

Fig. 6 Reward during online test for Scenario 3 (Threshold Loss). In this case during test URLLC traffic demand is higher, which induces more losses because of the threshold penalty. Even if global utility is lower because of heavier puncturing, the learning problem is easier to approximate.

in general challenging, if possible, to solve directly. URLLC and eMBB coexistence implies a downgrading of eMBB communications by reassigning eMBB users' resources to URLLC users in order to satisfy URLLC's demand. We solve the optimization problem arising from the aforementioned coexistence, focusing on maximizing a throughput based utility for eMBB users all while satisfying URLLC demands.

The effectiveness of the proposed approach was demonstrated through a variety of simulations. We used different scenarios and established realistic settings to prove the framework's performance, achieving results above current state of the art solutions. We were able to approximate online optimal solutions, even in non-convex scenarios in which state of the art techniques do not approximate well the optimal solution. On all scenarios our agents perform better than compared heuristics, achieving up to a 50% increase on eMBB users' utility with regards to well known state of the art proposals.

In future works we plan to analyze other learning algorithms, modifying restrictions and developing feature engineering, in order to find even better policies approximations. It would also be interesting to explore on implementing a combined training of both learning agents, by introducing in the loss function a utility divergence penalty. This may allow to exploit domain knowledge in order to achieve better results on particular problems. Finally, application of this statistical learning method over a broader series of problems with strong restrains and several timescales, as energy management, would be interesting in order to further validate the proposal.

References

[1] Kaur, J., Khan, M.A., Iftikhar, M., Imran, M., Emad Ul Haq, Q.: Machine learning techniques for 5g and beyond. IEEE Access **9**, 23472–23488 (2021). https://doi.org/10.1109/ACCESS.2021.3051557

[2] Morocho-Cayamcela, M.E., Lee, H., Lim, W.: Machine learning for 5g/b5g mobile and wireless communications: Potential, limitations, and future directions. IEEE Access **7**, 137184–137206 (2019)

[3] Klautau, A., Batista, P., González-Prelcic, N., Wang, Y., Heath, R.W.: 5g mimo data for machine learning: Application to beam-selection using deep learning. In: 2018 Information Theory and Applications Workshop (ITA), pp. 1–9 (2018). IEEE

[4] Huang, H., Guo, S., Gui, G., Yang, Z., Zhang, J., Sari, H., Adachi, F.: Deep learning for physical-layer 5g wireless techniques: Opportunities, challenges and solutions. IEEE Wireless Communications **27**(1), 214–222 (2019)

[5] Hasan, M.K., Shahjalal, M., Islam, M.M., Alam, M.M., Ahmed, M.F., Jang, Y.M.: The role of deep learning in noma for 5g and beyond communications. In: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 303–307 (2020). IEEE

[6] Li, J., Liu, M., Xue, Z., Fan, X., He, X.: Rtvd: A real-time volumetric detection scheme for ddos in the internet of things. IEEE Access **8**, 36191–36201 (2020)

[7] Wang, Y., Li, P., Jiao, L., Su, Z., Cheng, N., Shen, X.S., Zhang, P.: A data-driven architecture for personalized qoe management in 5g wireless networks. IEEE Wireless Communications **24**(1), 102–110 (2016)

[8] Martin, A., Egaña, J., Flórez, J., Montalbán, J., Olaizola, I.G., Quartulli, M., Viola, R., Zorrilla, M.: Network resource allocation system for qoe-aware delivery of media services in 5g networks. IEEE Transactions on Broadcasting **64**(2), 561–574 (2018). https://doi.org/10.1109/TBC.2018.2828608

[9] Ji, H., Park, S., Yeo, J., Kim, Y., Lee, J., Shim, B.: Ultra-reliable and low-latency communications in 5g downlink: Physical layer aspects. IEEE Wireless Communications **25**(3), 124–130 (2018)

[10] 3GPP: Etsi tr 138 912 5g; study on new radio (nr) access technology. ETSI **version 15.0.0**(Release 15) (2018)

[11] Popovski, P., Trillingsgaard, K.F., Simeone, O., Durisi, G.: 5g wireless

network slicing for embb, urllc, and mmtc: A communication-theoretic view. IEEE Access **6**, 55765–55779 (2018)

[12] Pedersen, K., Pocovi, G., Steiner, J., Maeder, A.: Agile 5g scheduler for improved e2e performance and flexibility for different network implementations. IEEE Communications Magazine **56**(3), 477–490 (2018)

[13] Pedersen, K., Pocovi, G., Steiner, J., Maeder, A.: Agile 5g scheduler for improved e2e performance and flexibility for different network implementations. IEEE Communications Magazine **56**(3), 210–217 (2018). https://doi.org/10.1109/MCOM.2017.1700517

[14] Pocovi, G., Pedersen, K.I., Mogensen, P.: Joint link adaptation and scheduling for 5g ultra-reliable low-latency communications. IEEE Access **6**, 28912–28922 (2018). https://doi.org/10.1109/ACCESS.2018.2838585

[15] Esswie, A.A., Pedersen, K.I.: Multi-user preemptive scheduling for critical low latency communications in 5g networks. In: 2018 IEEE Symposium on Computers and Communications (ISCC), pp. 00136–00141 (2018). https://doi.org/10.1109/ISCC.2018.8538471

[16] Bairagi, A.K., Munir, M.S., Alsenwi, M., Tran, N.H., Hong, C.S.: A matching based coexistence mechanism between embb and urllc in 5g wireless networks. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing. SAC '19, pp. 2377–2384. Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3297280.3297513. https://doi.org/10.1145/3297280.3297513

[17] Alsenwi, M., Tran, N.H., Bennis, M., Pandey, S.R., Bairagi, A.K., Hong, C.S.: Intelligent resource slicing for embb and urllc coexistence in 5g and beyond: A deep reinforcement learning based approach. IEEE Transactions on Wireless Communications **20**(7), 4585–4600 (2021). https://doi.org/10.1109/TWC.2021.3060514

[18] Elsayed, M., Erol-Kantarci, M.: Ai-enabled radio resource allocation in 5g for urllc and embb users. In: 2019 IEEE 2nd 5G World Forum (5GWF), pp. 590–595 (2019). https://doi.org/10.1109/5GWF.2019.8911618

[19] Almekhlafi, M., Arfaoui, M.A., Elhattab, M., Assi, C., Ghrayeb, A.: Joint resource allocation and phase shift optimization for ris-aided embb/urllc traffic multiplexing. IEEE Transactions on Communications **70**(2), 1304–1319 (2022). https://doi.org/10.1109/TCOMM.2021.3127265

[20] Anand, A., De Veciana, G., Shakkottai, S.: Joint scheduling of urllc and embb traffic in 5g wireless networks, 1970–1978 (2018). https://doi.org/10.1109/INFOCOM.2018.8486430

[21] Otsuka, H., Tian, R., Senda, K.: Transmission performance of an ofdm-based higher-order modulation scheme in multipath fading channels. Journal of Sensor and Actuator Networks **8**, 19 (2019). https://doi.org/10.3390/jsan8020019

[22] Carpentier, P., Chancelier, J.-P., de Lara, M., Rigaut, T.: Algorithms for two-time scales stochastic optimization with applications to long term management of energy storage (2019). working paper or preprint

[23] Xu, Y., Gui, G., Gacanin, H., Adachi, F.: A survey on resource allocation for 5g heterogeneous networks: Current research, future trends, and challenges. IEEE Communications Surveys 'I&' Tutorials **23**(2), 668–695 (2021). https://doi.org/10.1109/COMST.2021.3059896

[24] Zhang, D., Qiao, Y., She, L., Shen, R., Ren, J., Zhang, Y.: Two time-scale resource management for green internet of things networks. IEEE Internet of Things Journal **6**(1), 545–556 (2019). https://doi.org/10.1109/JIOT.2018.2842766

[25] Chen, T., Zhang, X., You, M., Zheng, G., Lambotharan, S.: A gnn-based supervised learning framework for resource allocation in wireless iot networks. IEEE Internet of Things Journal **9**(3), 1712–1724 (2022). https://doi.org/10.1109/JIOT.2021.3091551

[26] Bao, Z., Zhou, Q., Yang, Z., Yang, Q., Xu, L., Wu, T.: A multi time-scale and multi energy-type coordinated microgrid scheduling solution—part i: Model and methodology. IEEE Transactions on Power Systems **30**(5), 2257–2266 (2015). https://doi.org/10.1109/TPWRS.2014.2367127

[27] Liu, Z., Wu, Q., Ma, K., Shahidehpour, M., Xue, Y., Huang, S.: Two-stage optimal scheduling of electric vehicle charging based on transactive control. IEEE Transactions on Smart Grid **10**(3), 2948–2958 (2019). https://doi.org/10.1109/TSG.2018.2815593

[28] Bengio, Y., Lodi, A., Prouvost, A.: Machine learning for combinatorial optimization: A methodological tour d'horizon. European Journal of Operational Research **290**(2), 405–421 (2021). https://doi.org/10.1016/j.ejor.2020.07.063

[29] Eisen, M., Zhang, C., Chamon, L.F.O., Lee, D.D., Ribeiro, A.: Learning optimal resource allocations in wireless systems. IEEE Transactions on Signal Processing **67**(10), 2775–2790 (2019). https://doi.org/10.1109/TSP.2019.2908906

[30] Aitchison, J.: The statistical analysis of compositional data. Journal of the Royal Statistical Society. Series B (Methodological) **44**(2), 139–177 (1982). Accessed 2023-07-03

[31] Zhang, Q., Kassam, S.A.: Finite-state markov model for rayleigh fading channels. IEEE Transactions on Communications **47**(11), 1688–1692 (1999). https://doi.org/10.1109/26.803503

[32] Diamond, S., Boyd, S.: CVXPY: A Python-embedded modeling language for convex optimization. Journal of Machine Learning Research **17**(83) (2016)

[33] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[34] Chollet, F., et al.: Keras. https://keras.io (2015)

Declarations

This work is partially funded by the Agencia Nacional de Investigación e Innovación (ANII) project 'Artificial Intelligence for 5G networks' (FMV_1_2019_1_155700). Martín Randall's PhD on which this article is inscribed has the support of a scholarship granted by the Agencia Nacional de Investigación e Innovación (ANII).

The authors have no relevant financial or non-financial interests to disclose.

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Martín Randall, supported by Gonzalo Belcredi, and advised by Pablo Belzarena and Federico Larroca. The first draft of the manuscript was written by Martín Randall, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

All data generated and code can be found at https://gitlab.fing.edu.uy/ai45g/puncturing-ai45g.