# Intelligent Multi-Branch Allocation of Spectrum Slices for Inter-Numerology Interference Minimization*

Marco Zambianco[a,*], Giacomo Verticale[a]

[a] *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, piazza Leonardo da Vinci 32, 20133 Milano, Italy*

**Abstract**

Network slicing and mixed-numerology access schemes cover a central role to enable the flexible multi-service connectivity that characterizes 5G radio access networks (RAN). However, the interference generated by the simultaneous multiplexing of radio slices having heterogeneous subcarrier spacing can hinder the isolation of the different slices sharing the RAN and their effectiveness in meeting the application requirements. To overcome these issues, we design a radio resource allocation scheme that accounts for the inter-numerology interference and maximizes the aggregate network throughput. To overcome the computationally complexity of the optimal formulation, we leverage deep reinforcement learning (DRL) to design an agent capable of approximating the optimal solution exploiting a model-free environment formulation. We propose a multi-branch agent architecture, based on Branching Dueling Q-networks (BDQ), which ensures the agent scalability as the number of spectrum resources and network slices increases. In addition, we augment the agent learning performance by including an action mapping procedure designed to enforce the selection of feasible actions. We compare the agent performance to several benchmarks schemes. Results show that the proposed solution provides a good approximation of the optimal allocation in most scenarios.

*Keywords:* Network slicing, Reinforcement learning, Interference

## 1. Introduction

Network slicing makes it possible to overlay multiple virtual networks on a common physical network infrastructure. Every network slice employs a pool of heterogeneous network resources, ranging from core network resources to Radio Access Network (RAN) resources, in order to efficiently tailor to a specific user application needs, expressed as a set of Service Level Agreement (SLA) requirements [1]. In this context, the policy for partitioning resources among coexisting network slices is responsible for providing isolation among slices so as to give the illusion that they are logically independent one from the other. Inter-slice isolation is particularly important on the radio interface due to the fact that, unlike computational resources (e.g. CPUs), radio resources are often affected by external interference sources that negatively affect their service provisioning capability. In addition, spectrum is a limited and inherently dynamic resource that depends on the radio propagation environment as well as the user mobility statistics. For this reason, an effective spectrum allocation policy, on one hand, has to enforce inter-slice isolation by limiting the inter-slice interference and, on the other hand, has to ensure a suitable scheduling of the radio resources to accommodate each network slice service request.

To efficiently satisfy these requirements, it is important to consider the impact of the allocation policy over the common physical layer structure upon which the shared spectrum resources are mapped. In this regard, mixed-numerologies schemes are proposed as medium access schemes in 5G networks to boost the system flexibility. Differently from the classical orthogonal division multiple access (OFDMA), mixed-numerologies schemes allow to multiplex different subcarriers having a heterogeneous subcarrier spacing within the same OFDM symbol [2]. Specifically, every fixed frequency-time slot, also denoted as resource block (RB), can be assigned with a different numerology that dictates the granularity level of the subcarrier spacing. The advantage of this technique makes it easier to overcome different radio propagation characteristics as the channel frequency selectivity can be effectively counteracted by a suitable subcarrier spacing. However, the loss of orthogonality between contiguous RBs, that are simultaneously multiplexed on the common RB grid, produces inter-numerology interference (INI) that can substantially hinder the transmission performance [3].

However, the majority of the current research activity has investigated the RAN slicing resource allocation and the INI minimization topics separately. For these reason, we jointly consider the aforementioned problems and we design a suitable allocation scheme that maximizes the aggregated throughput of each network slices and, at the same time, mitigates the INI. To better illustrate the

*This work has been partially funded by Regione Lombardia project BASE-5G – Broadband Interfaces and services for Smart Environments enabled by 5G technologies, project number 1155850

*Corresponding author

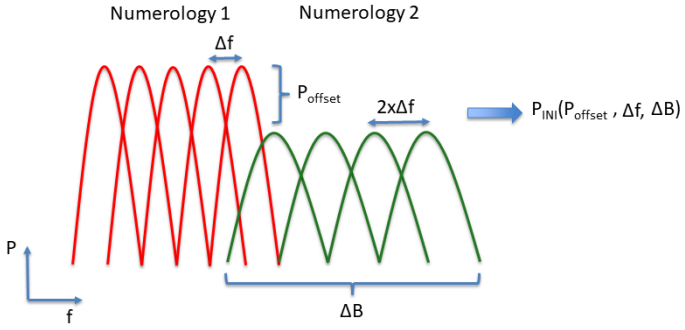



Figure 1: Inter-numerology interference relationship with physical layer parameters when two spectrum slices of different numerologies are contiguously allocated.

considered problem, in Fig. 1 we provide a scheme that qualitatively shows the INI dynamic between two different numerologies, but it can be easily generalized if more numerologies are considered.

The INI power depends on several factor like the subcarrier spacing difference between contiguous numerolgoies, the bandwidth allocated to each numerology and the power difference between the subcarriers. Specifically, the higher are the aforementioned quantities, the higher is the INI power that is mutually generated across the different numerologies. Consequently, other than employing specific interference cancellations techniques at the physical layer, a suitable spectrum allocation can effectively reduce the INI impact by prioritizing an allocation that reduces the subcarrer power offset difference as well as their numerology gap.

We tackle the discussed problem using deep reinforcement learning (DRL). This framework has recently been applied in different wireless problems with promising results [4] [5]. In detail, we design a DRL agent that infers the relationship between the INI power and the wireless channel fading dynamic to maximize the aggregated network slice throughput. Note that, differently from our previous work in [6], we propose an agent based on BDQ networks that efficiently scales with the number of spectrum resources and network slices. More precisely, the main contribution of this work are:

- We formulate a integer non-convex optimization problem that allocates the available spectrum resources to multiple network slices having different numerologies in order to maximize the cumulative throughput. The designed objective function directly embeds the analytical expression of the INI, which allows to reliably compute the optimal allocation according to the wireless channel status of users.

- We address the computational complexity of the optimal formulation by designing a DRL agent based on a multi-branch network architecture. Specifically, each network branch is in charge of the allocation of a subset of spectrum resources. The coordination between the different branches is ensured by a shared module that provides a common representation of the environment status.

- We design an action mapping scheme that ensures the feasibility of every action selected by the agent. As a matter of fact, it is not possible to naturally enforce the constraints that characterize the optimal formulation, thus we propose this approach in order to boost the agent convergence performance to the optimal policy.

- We assess the agent performance in terms of scalability performance and optimal solution approximation. Results show that the agent successfully converges under different combinations of network configurations and it provides a good approximation of the optimal solution in most scenarios. In addition, it provides better gains than the single-branch agent counterpart when the most complex system scenarios are assessed [6].

The remainder of the paper is organized as follows. We present the related work in Section 2. We describe the system model in Section 3. We discuss the optimal problem formulation as well as the agent environment definition in Section 4. We present the multi-branch agent architecture and the related training process in Section 5. We show the simulation results in Section 6. Finally, the conclusion is drawn in Section 7.

## 2. Related work

In the context of 5G radio resource allocation, deep reinforcement learning has been recently proposed as resolution scheme to overcome the complexity derived from classical model-based approaches [7]. However, many of the proposed solutions consider specific wireless applications such as vehicular networks [8]. DRL agents in this field are designed to be employed by each vehicular user in order to mitigate the interference raised by an uncoordinated dynamic spectrum access [9]. For this reason, their effectiveness is limited when they are used for spectrum allocation performed on generic cellular networks composed by multiple users requiring different services. Supported by this observation, we consider the application of DRL for a multi-user scenario in a mixed-numerology 5G RAN shared by different network slices. Due to the combinatorial nature of the considered allocation problem, we leverage a multi-branch agent architecture that ensures the agent scalability as the number of resources and/or network slices increases.

A general overview of the basic concepts of RAN slicing and mixed-numerology access schemes as well as their main challenges can be found in [10] [11]. We can classify the recent work on these topics in three main categories: INI-agnostic RAN slicing schemes, that provide spectrum

allocation policies without accounting for the INI between different slices, INI-aware RAN slicing schemes, that consider the INI impact on the resource allocation process, and finally INI cancellation schemes that mitigate the INI effect by means of signal processing techniques performed at the physical layer.

### *2.1. INI-agnostic RAN slicing schemes*

The works in [12] and [13] provide an application of DRL in the context of resource allocation in RAN slicing. The agents are designed to learn an allocation of the radio resources among different network slices that can accommodate heterogeneous QoS requirements. However, since authors does not consider a mixed-numerology access scheme, the performance analysis remains uncovered when a mixed-numerology resource grid is employed. Differently, the authors of [14] design multiple scheduling algorithms to accommodate low latency and multi-broadband users at heterogeneous time granularity on a common physical layer. Similarly, the work in [15] leverages the transmission frame flexibility provided by mixed-numerology schemes to design a self-adaptive transmission time interval scheduling strategy for low latency and multi-broadband services. However, although [14] and [15] consider a mixed-numerology access scheme, the INI impact on the algorithms performance is not included. The authors of [16] propose a multi-agent DRL framework that assigns radio resources according to the slice service requirements without over-provisioning the available RAN spectrum. Differently from this work, we assume that the number of radio resources required by each network slices is already provided and we instead address the problem of multiplexing mixed-numerology spectrum slices on a shared physical layer by actively modeling the wireless channel behavior as well as the INI within the agent formulation.

### *2.2. INI-aware RAN slicing schemes*

The authors of [17] and of [18] design an INI mitigation scheduling algorithm based on adaptive guard intervals and on subband power offset reduction between different numerologies, respectively. However, the proposed schemes do not ensure an optimal INI minimization as they do not analytically consider the INI within the problem formulation. The authors of [19] design a max-min Knapsack problem that allocates the available spectrum slices in order to accommodate the users latency requirements. The proposed scheduling scheme also considers the INI affecting each slice. However, the INI behavior is approximated by neglecting the contribution of numerologies with a small subcarrier spacing. In addition, only the large-scale fading is considered by the optimization procedure. Differently from these works, we propose an optimal INI-aware spectrum allocation scheme that leverages an analytical INI estimation and that accounts for the small-scale fading dynamic of the users. Finally, our previous work [6] proposes a DRL agent that multiplexes spectrum slices of different numerologies employing an optimal INI-aware reward function formulation. However, this approach is unpractical when the number of resources and numerologies increases due to the fact that the agent requires the enumeration of every feasible allocation to approximate the optimal policy. We overcome this issue by proposing an alternative agent architecture that does not have such limitation and we design an action mapping module to guarantee the action feasibility.

### *2.3. INI cancellation schemes*

In [20], a novel transreceiver design is proposed to reduce the INI in mixed-numerology systems. The authors design a low-complexity encoding and decoding procedure that lowers the interference energy variance by uniformly spreading the INI across different subcarriers. The authors of [21] combine filter-OFDM together with index modulation, that allows to activate only a subset of the available subcarriers, to mitigate the INI. The effectiveness of the considered approach is quantified by a lower decoding error probability. The authors of [22] improve the spectral efficiency of mixed-numerology schemes by designing a novel guard band insertion mechanism. The latter leverages the specific INI pattern affecting the various subcarriers in order to reduce the guard band size between different numerologies. The authors of [23] design a precoding algorithm for INI mitigation that is applied at the transmitter side. The proposed scheme ensures a lower interference level for edge subcarriers which in turn improves the decoding performance capability at the receiver side. The aforementioned works [20]-[23] provide different signal processing techniques to minimize the INI which also accounts for the wireless channel fading. However, they focus on the link level performance when subbands of two different numerologies are considered. Differently, we provide an INI mitigation approach from a resource allocation perspective where multiple numerologies are dynamically allocated over a shared spectrum and the multi-user diversity is exploited to boost the system throughput.

## 3. System Model

The system model is mainly composed by two parts, the RAN slicing architecture that defines how different network slices share the radio interface and the INI model that allows to quantitatively measure the impact of a heterogeneous resource grid on users belonging to different network slices.

### *3.1. RAN slicing architecture*

We consider a RAN whose physical infrastructure is administrated by the network owner (NO). The latter manages the assignment of the spectrum resources between $M$ mobile virtual network operators (MVNO) sharing the

radio interface in order to guarantee the provisioning feasibility of the different network services. We assume that each MVNO manages a logically independent network slice that schedules the assigned radio resources to its own users based on the SLA requirements dictated by the application. Let $U_m$ be the users of the $m$th MVNO. Consequently, the system can be viewed as a two-layer radio resource scheduler. The upper layer is managed by the individual schedulers of the various MVNOs, whereas the lower layer is identified by the spectrum assignment policy of the NO, which accommodates the RAN spectrum among the MVNOs. In other words, the NO can be considered as a "network slice scheduler". In Fig. 2, we schematically depict the considered RAN slicing architecture.

A mixed-numerology OFDMA scheme is employed at the physical layer, where we split the frequency-selective channel in $K$ independent flat-fading subchannels of bandwidth $W$. Using 5G terminology, we can say that every subchannel is the equivalent to a bandwidth part (BWP), which is a subset of contiguous resource blocks (RB) having the same numerology [24]. Each MVNO $m$ employs a different numerology type, $\mu_m$, in order to effectively accommodate the heterogeneous radio propagation behaviors of its users and to improve the transmission performance. The numerology type defines the subcarrier spacing within each RB composing the different subchannels. Formally, a RB belonging to the MVNO $m$ is characterized by a subcarrier spacing of $\Delta f_m = 15\,\text{kHz} \cdot 2^{\mu_m}$ with $0 \leq \mu_m \leq 4$ as defined in the 3GPP NR specification [25]. The NO has perfect channel state information (CSI) of every user regardless from the MVNOs to which it belongs. Moreover, it periodically computes the number of subchannels required by each MVNO in order to accommodate their own user service demand by following the spectrum assignment policy $S_m$. The latter is formally defined as

$$S_m = \left\{ \ell_m, m \in M : \sum_{m \in M} \ell_m = K \right\},$$

where $\ell_m$ indicates the number of subchannels that are assigned to MVNO $m$. Note that we do not focus on how $S_m$ is computed since our goal is the multiplexing of the subchannels assigned to the various MVNOs on the shared RAN spectrum. For this reason, we consider that $S_m$ is provided as input parameter by the network and it can be computed by relying on prediction techniques that forecast the required spectrum usage of each slice based on the related historical service provisioning. We remark that the latter also indirectly accounts for the experienced INI level since it affects the overall data rate. Consequently, it is reasonable to assume that $S_m$ provides a reliable INI-aware subchannel assignment policy that is further enforced by the proposed INI-aware subchannel multiplexing policy. Given a user $u$, an MVNO $m$, and a subchannel $k$, we model the channel fading as composed by a slow fading component, $\alpha_m^u$, and a fast fading component, $h_m^u(k)$. The former accounts for the path loss and shadowing effects, whereas the latter accounts for the multi-path effect. Analytically, the power gain of subchannel $k$ computed by user $u$ belonging to MVNO $m$ can be expressed as $g_m^u(k) = \alpha_m^u h_m^u(k)$, where $h_m^u(k)$ is modeled as an exponentially distributed random variable with unit mean.

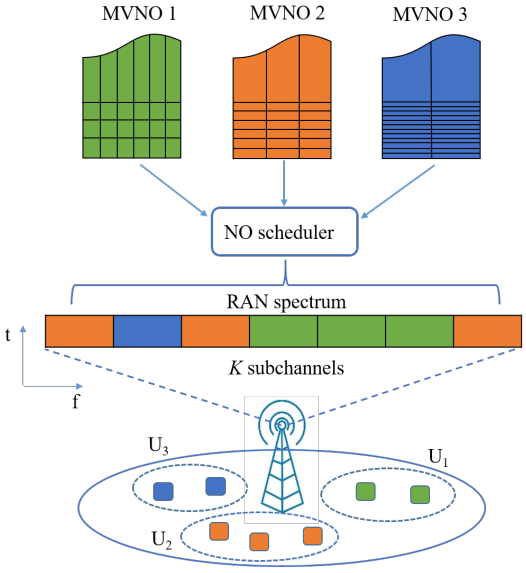



Figure 2: RAN slicing architecture. The NO multiplexes the active MVNOs on the available subchannels composing the RAN spectrum.

### 3.2. INI analytical model

The simultaneous allocation of subchannels composed by RBs of heterogeneous numerologies generates INI. The orthogonality condition across subcarriers is only guaranteed when the adopted numerology is homogeneous. To quantitatively model the INI power dynamic, we rely on the analytical formulation proposed by authors in [3], which we adapted to fit the logical subchannel division of the RAN spectrum. In details our formulation characterizes the INI power over the fixed number of subcarriers that is defined by the subchannel numerology. Given two subchannels, $k$ and $k'$, user $u$ is on subchannel $k$ using a numerology with subcarrier spacing $\Delta f_m$, while another user is on subchannel $k'$ using a numerology with subcarrier spacing $\Delta f_{m'}$, then the INI power affecting user $u$ can be computed as, if $\Delta f_m < \Delta f_{m'}$,

$$I_m^u(k,k') \approx \frac{P_T(k')}{N_{k'}} \sum_{z=1}^{N_k} \sum_{v=1}^{N_{k'}} \frac{g_m^u(k')}{N_{k'} N_k} \left[ \left| \frac{\sin[\frac{\pi}{N_k} w(z,v) \xi N_{k'}^{(T)}]}{\sin(\frac{\pi}{N_k} w(z,v))} \right|^2 + \xi \left| \frac{\sin[\frac{\pi}{N_k} w(z,v) N_{k'}^{(T)}]}{\sin[\frac{\pi}{N_k} w(z,v)]} \right|^2 \right], \quad (1)$$

otherwise, if $\Delta f_m > \Delta f_{m'}$, as

$$I_m^u(k,k') \approx \frac{P_T(k')}{N_{k'}} \sum_{z=1}^{N_k} \sum_{v=1}^{N_{k'}} \frac{g_m^u(k')}{N_{k'} N_k} \left| \frac{\sin[\frac{\pi}{N_{k'}} w(z,v) N_k]}{\sin[\frac{\pi}{N_{k'}} w(z,v)]} \right|^2 \tag{2}$$

where $N_k = W/\Delta f_m$ corresponds to the number of subcarriers in subchannel $k$, $P_T(k)$ is the power allocated on subchannel $k$, $N_k^{(T)} = N_k + N_k^{CP}$ denotes the total number of subcarriers in subchannel $k$ with $N_k^{CP}$ defining the number of subcarriers employed as cyclic-prefix, $\xi = \lfloor N_k / N_{k'}^T \rfloor$ is the number of overlapping OFMD symbols within the same transmission frame, and $w(z,v)$ is the spectral distance between subcarriers of different numerolgies and it is calculated as the total number of subcarriers separating subcarrier $z$ from subcarrier $v$.

From (1) and (2), we observe that INI power mainly depends on the spectral distance between subcarriers of different numerologies and the power allocated to each subcarrier. More specifically, the interference suffered from each subcarrier increases as the numerology gap with respect to subcarriers using a wider subcarrier spacing increases. Similarly, the higher is the power allocated to each subcarrier, the higher is the interference generated on neighbor subcarriers.

We quantify the subchannel quality measured by each user $u$ belonging to MVNO $m$ and associated to each subchannel $k$ as the signal-to-interference-plus-noise ratio (SINR)

$$\gamma_m^u(k) = \frac{P_T(k) g_m^u(k)}{\sigma_w^2 + \sum\limits_{m' \neq m} \sum\limits_{k' \neq k} x_{m',k'} I_m^u(k,k')}, \tag{3}$$

where $x_{m,k}$ is the binary subchannel allocation indicator that has value $x_{m,k} = 1$ if subchannel $k$ is allocated to MVNO $m$ and $x_{m,k} = 0$ otherwise, and $\sigma_w^2$ is the white Gaussian noise power over each subchannel.

From (3), we note that the INI power on subchannel $k$ is related to the subchannel gains of the remaining subchannels having different numerologies. Specifically, the higher is the power on adjacent subchannels of different numerologies, the higher is the INI power generated on subchannel $k$. Intuitively, exploiting the fading independence over the frequency domain across the different subchannels, each user can reduce the INI by accessing subchannels with high gains that are surrounded by subchannels having a much lower gain. In other words, an INI-aware subchannel allocation that leverages the multiplexing gains provided by the wireless channel can mitigate the INI impact affecting the various users. We formalize this observation as an optimization problem in the next section.

## 4. Problem formulation

In this section, we formulate the problem for the optimal spectrum allocation and we formalize the system environment that is used by the DRL agent to compute an allocation policy approximating the optimal solution.

### *4.1. Optimal resource allocation*

Based on the RAN slicing architecture described beforehand, the ideal subchannel allocation that maps the various subchannel over the common RAN spectrum according to $S_m$ should maximize the aggregated MVNO throughput performance, which means that each MVNO should have access to the subset of subchannels that ensures the highest data rate to its users.

Stemming from this requirement, we formalize the resource allocation problem as

$$\max_x \sum_{m=1}^{M} \sum_{k=1}^{K} \frac{1}{U_m} \sum_{u=1}^{U_m} x_{m,k} \cdot W \log_2 (1 + \gamma_m^u(k)) \tag{4}$$

subject to

$$\sum_{k=1}^{K} x_{m,k} = \ell_m \quad \forall m \in M \tag{5}$$

$$\sum_{m=1}^{M} x_{m,k} \leq 1 \quad \forall k \in K \tag{6}$$

$$x_{m,k} \in \{0,1\} \quad \forall m \in M, \forall k \in K. \tag{7}$$

Problem constraints ensure the feasibility of resource allocation according to the proposed system model. In detail, (5) guarantees that the number of subchannels allocated to each MVNO follows the spectrum assignment policy $S_m$. Equation (6) makes sure that each subchannel is allocated to a single MVNO only. Finally, (7) enforces the integer nature of the problem through the binary optimization variable $x_{m,k}$.

The integrity condition (7) on the optimization variable makes the solution computation challenging. Moreover, the object function is generally non-convex due to the fact that it can be seen as a difference of convex functions. This makes the proposed problem formulation NP-hard [26], hence (4) is unpractical for real systems that perform the allocation of the radio resources within a very strict amount of time at the physical layer level. We leverage the DRL theory to train an agent that can quickly compute a suitable allocation policy approximating the optimal solution.

### *4.2. Environment formulation*

DRL provides an iterative method to compute an optimal policy for solving a Markov Decision Process (MPD), where the transition probabilities from each state towards other states are unknown [27]. Formally, an MDP can be formulated as the 5-tuple $(S, A, p(s'|s,a), R(s,a), \gamma)$, where $S$ and $A$ denotes, respectively, the state space and action space, $p(s'|s,a)$ denotes the transition probability from state $s$ at time $t$ toward state $s'$ at time $t+1$ and depends on the current state $s$ and the action $a$, $R(s,a)$

is the immediate reward that is obtained by performing action $a$ under state $s$, that is discounted over time by a factor $\gamma$, with $0 \leq \gamma < 1$. The goal of the learning is to find the optimal policy that allows to maximize the expected discounted reward from any initial state $s$, i.e. $\sum_{i=0}^{\infty} \gamma^i R_{t+1+i}$. We present an MDP formulation of the original system model by means of a set states $S$, which can be explored according to the action space $A$ in order to maximize the designed reward function $R$.

*State space.* The environment is characterized by the subchannel gains $g_m^u(k)$ and the INI power measured before the attenuation introduced by the multi-path fading channel. In other words, this is the INI power generated at the base station according to the selected subchannel allocation and is analytically computed by setting $g_m^u(k) = 1, \forall u \in U_m$, i.e. without considering the fading gain, in (1) and (2) . The motivation behind this design choice is to emphasize within the state space representation the difference between the subchannel fading, that is purely stochastic, and the INI power that is strictly related to the selected subchannel allocation. Such representation allows the agent to better discriminate the impact of the performed action on the environment due to the highlighted deterministic component. At each time slot $t$, the agent samples the environment and observes the state $s_t$ that belongs to the state space $S$,

$$S = \{G[k], I[k]\}_{k \in K} \tag{8}$$

where

$$G[k] = \{g_m^u(k)\}_{m \in M, u \in U_m}, \tag{9}$$

$$I[k] = \Big\{ \sum_{m' \neq m} \sum_{k' \neq k} x_{m',k'} I_m(k,k') \Big\}_{m \in M}. \tag{10}$$

Note that the term $I_m(k,k')$ in (10) is the same as the term $I_m^u(k,k')$ in (1) and (2). We dropped the index $u$ since (10) provides the same value for the users belonging to the same MVNO. According to this environment representation, the state space dimension has size $K \cdot (\sum_m U_m + 1)$, hence it scales linearly with the total number of users and subchannels.

*Action space.* The action space is composed by all the possible subchannel allocations that can be computed according to the number of active MVNOs. We represent each action as a $K$-dimensional vector $\mathbf{a}_t$:

$$\mathbf{a}_t = (a_1, ..., a_K) \tag{11}$$

where the generic coordinate $a_k$, with $1 \leq a_k \leq M$ indicates that subchannel $k$ is allocated to MVNO $m$. According to this action space formulation, the total number of actions composing the action space $A$ is equal to $M^K$. There are two issues with this representation. First, the exponential increase of the available actions can severely affect the agent learning performance due the large action space dimensionality that hinders the environment exploration. Second, a subset of actions is not feasible due to the fact that constraint (5) is not naturally enforced within the action space. In the next section, we address these drawbacks by designing an agent that can efficiently manages the large action space while simultaneously ensuring action feasibility.

*Reward function.* We directly employ the objective function (4) to model the reward obtained by the agent at each time step. Consequently, the agent is going to maximize the obtainable reward over time, thus approaching an expected reward value close to the optimal one. At every time step, the agent evaluates the effectiveness of the selected action by observing the reward $R_t$ computed as (4) relying on the state formulation (8). As a matter of fact, the subchannel gains of each user are used to compute the data rate according to the selected subchannel allocation which also provides the information to calculate the INI generated by the corresponding MVNO multiplexing.

## 5. Multi-branch resource allocation agent

We now present the DRL agent, referred as multi-branch resource allocation (MBRA) agent, to compute a suitable approximation of the optimal subchannel allocation. Since the agent architecture can be considered as an extension of deep Q-learning to multi-dimensional action spaces, we first provide a brief overview of its core elements, then we present the enhancement required to support the scalability in large action spaces.

### *5.1. Deep Q-learning general structure*

Q-learning is an off-policy reinforcement learning scheme that allows to compute the optimal policy under a model-free environment formulation [28]. The optimal policy $\pi^*$ is computed by taking the maximum value of the Q-function at every time step. The latter is defined as the average discounted reward obtainable starting from state $s$, taking action $a$ and following the policy $\pi$. Formally, we can write the optimal policy as

$$Q_{\pi^*}(s,a) = \max_{a \in A} Q_\pi(s,a) \tag{12}$$

where

$$Q_\pi(s,a) = E_\pi[\sum_{i=0}^{\infty} \gamma^i R_{t+1+i} | S_t = s, A_t = a]. \tag{13}$$

Since the Q-function is unknown at the beginning of the training phase, the agent iteratively approximates its value by means of subsequent Temporal-Difference (TD) updates that consist of a weighted average between the old Q-function value obtained at time step $t$ and the new Q-function value obtained in the next time step $t+1$. Analytically, at every time step, $Q(s,a)$ is updated as

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha[R(s_t, a_t) + \gamma \max_{a \in A} Q(s_{t+1}, a)], \quad (14)$$

where $\alpha \in [0, 1]$ is the learning rate.

Classical Q-learning employs tables as data structure to individually store the Q-function values associated to each action-state pair. However, when the number of states and/or actions grows large, the agent convergence performance is hindered as many state-action pairs are rarely visited, thus resulting in a sporadic update of the related Q-function values. To overcome this issues, Q-learning evolved into deep Q-network (DQN) by introducing deep neural networks (DNN) to boost the learning process. This technique allows to approximate the Q-function by means of a suitable DNN with weights $\{\boldsymbol{\theta}\}$.

The DNN weights $\{\boldsymbol{\theta}\}$ are updated in order to minimize the loss, $L(\boldsymbol{\theta})$, between the Q-function values computed on subsequent time steps. Formally, they are updated as

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}), \quad (15)$$

where $L(\boldsymbol{\theta})$ is defined as

$$L(\boldsymbol{\theta}) = \sum_{B} [R(s_t, a_t) + \gamma \max_{a \in A} Q(s_{t+1}, a; \boldsymbol{\theta}') - Q(s_t, a_t; \boldsymbol{\theta})]^2. \quad (16)$$

In (16), $\{\boldsymbol{\theta}'\}$ corresponds to the weights of a second DNN that is used to stabilize the Q-function computation convergence and it is updated as $\{\boldsymbol{\theta}' = \boldsymbol{\theta}\}$ every few time steps. Parameter $B$ is the size of the mini-batch that is randomly sampled from the experience-replay buffer. The latter collects and stores the most recent $N$ experience tuples $(s_t, a_t, s_{t+1}, r_{t+1})$ observed by the agent during the training phase. This sampling procedure improves the agent learning performance by providing uncorrelated experience tuples between subsequent weights updates [29].

### *5.2. Multi-branch Agent Architecture Overview*

In order to mitigate the scalability issues stemming from the growth of the action space, we adopt a Divide-and-Conquer strategy. Suppose that each subchannel is managed by an agent, with each agent selecting, for each time slot, the MVNO whose users achieve the highest SNR. The aggregated action, which consists of the union of the allocations performed by each agent, corresponds to the multi-dimensional vector $\mathbf{a}_t$ originally defined in (11).

If there is no INI, it is trivial to see that the above strategy finds the optimal solution. When INI is considered and no agent coordination is enforced, this strategy leads to poor results since every agent would act independently during the learning process, thus making the convergence process not stationary.

To overcome these challenges, we employ a DRL scheme denoted as Branching Dueling Q-network (BDQ) [30]. This agent leverages the Dueling Network architecture proposed by authors in [31] to distribute the representation of the state-action values across independent DNN branches, each one expressing the advantage function related to every sub-action (i.e. each coordinate of the multi-dimensional action $\mathbf{a}_t$). The advantage function quantifies how much better is the chosen sub-action with respect to the current state value estimation.

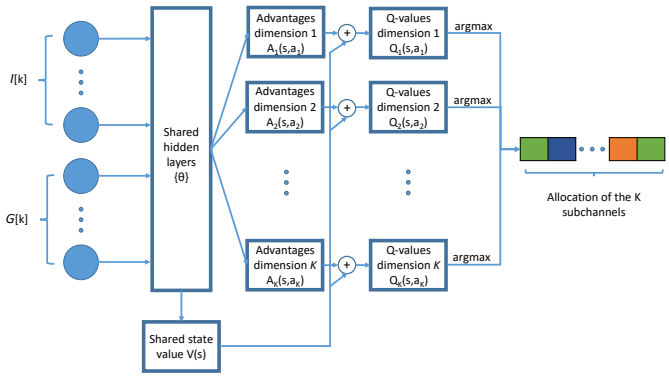



Figure 3: Multi-branch DNN architecture. Every network branch provides the Q-function value $Q_k(s, a_k)$ that is computed by aggregating the shared state value module with the advantage function of each sub-action $a_k \in \{1, ..., M\}$. The subchannel allocation is computed by taking the maximum Q-function value from each network branch.

The coordination among different network branches is ensured by a shared decision module that computes a common state value estimation of every observed state. The Q-function value of each sub-action is individually computed by means of an aggregation layer that combines the shared representation module with each single advantage function. In Fig. 3 we report a scheme of the discussed multi-branch architecture. Intuitively, every network branch can be considered as an independent agent that is in charge of the optimal policy computation according to the action space defined by the values of the $k$-th coordinate in $\mathbf{a}_t$.

Thanks to this scheme, we can observe a linear increase of the number of output neurons as the number of subchannels and/or MVNOs increases rather than the exponential increase that would occur if each subchannel allocation was treated as a scalar value. Quantitatively, the total number of output neurons required by the multi-branch architecture is $M \cdot K + 1$ which is lower than the action space dimensionality $M^K$. We now cover more in detail the main elements of the considered agent architecture.

*Shared state-value estimator.* This module represents a separated branch in the DNN representation and provides the estimate of the reward given the current state. In general, the state value function is directly related to the Q-function value and it can be retrieved by weighting its value with respect to the policy $\pi(a|s)$ that provides the

probability of taking the action $a$ given the state $s$, i.e.,

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) Q(s,a), \tag{17}$$

As mentioned earlier, the estimated value is used by the various network branches to coherently correlate their action choice according to the predicted achievable throughput in the next time slot.

*Dueling Network Architecture.* This network architecture makes it possible to efficiently estimate the Q-function without the need of individually inspecting the Q-function value related to each state-action pair. As a matter of fact, it is possible to accelerate the agent convergence performance by expressing the Q-function in terms of state value function $V(s)$ and advantage function $A(s,a)$ and aggregating the two modules as

$$Q(s,a) = V(s) + \Big[A(s,a) - \frac{1}{n} \sum_{a' \in A} A(s,a')\Big] \tag{18}$$

where $n$ is the number of available actions. This alternative Q-function value computation allows the agent to generalizing more rapidly among actions that provide similar rewards [31]. In our scenario, this feature is particular useful since many subchannels allocations are likely to provide comparable throughput values given different INI levels, hence a generalization of their relationship can assist the agent convergence.

### *5.3. Agent training*

We split the training procedure in two steps, the action selection phase and the action mapping phase. The first step computes the weights of the DNN associated to the MBRA agent, whereas the second step changes unfeasible actions selected by the agent during the exploration process into feasible actions. The overall training procedure is reported in Algorithm 1.

*Action selection phase.* To ensure the agent flexibility in adapting to multiple radio scenarios, the training procedure is composed by episodes having a different user distribution that is randomly generated at the beginning of every new episode, where the time step granularity of each episode is dictated by the CSI update performed by the users (lines 3-11). The agent aims at maximizing the Q-function value related to the sub-actions that are available in each network branch. To achieve this goal, the agent needs to explore the environment in order to learn new action-state pair combinations that can improve the obtainable reward as well as to exploit its knowledge about the environment in order to achieve the highest possible reward. This exploration/exploitation procedure is implemented by means of a $\epsilon$-greedy policy. Specifically, with probability $\epsilon$ the agent computes a subchannel allocation by randomly selecting a sub-action $a_k$ from each network

**Algorithm 1:** MBRA training procedure

**Input:** $K$, $M$, $U_m$, $S_m$, $\gamma$, $\alpha$
**Output:** Allocation of $K$ subchannels to $M$ MVNOs according to the spectrum assignment policy $S_m$

1 Initialize the wireless environment;
2 Initialize the agent DNN with $K$ network branches having $M$ output neurons each;
3 **foreach** *episode* **do**
4  **foreach** *MVNO* $m \in M$ **do**
5  Randomly place $U_m$ active users over the BS coverage area;
6  **end**
7  **foreach** *time step* **do**
8  **foreach** *subchannel* $k \in K$ **do**
9  Simulate INI power in every subchannel using (10);
10  Generate CSI reporting of every user $u \in U_m$;
11  **end**
12  Observe the environment state $s_t$;
13  Select action $\mathbf{a}_t$ with probability 1-$\epsilon$ according to (19) or with probability $\epsilon$ by randomly choosing $a_k$ in each branch;
14  **if** $\mathbf{a}_t$ *is unfeasible* **then**
15  Compute the aggregate action using (27) where each network branch is selected according to Alghorithm 2;
16  **end**
17  Store the experience tuple $(s_t, \mathbf{a}_t, r_{t+1}, s_{t+1})$ in the experience-replay buffer;
18  Sample a mini-batch of size $B$ following the probability distribution (23);
19  Update $\boldsymbol{\theta}$ in order to minimize $L(\boldsymbol{\theta})$ in (22) using the sampled mini-batch, where each TD error is weighted based on (25);
20  **end**
21 **end**

branch. Differently, with probability 1-$\epsilon$ the agent computes a subchannel allocation that is built by selecting from each network branch the sub-action $a_k$ providing the highest Q-function value (line 13). Note that the computed spectrum allocation is applied to every TTI occurring between two subsequent CSI reporting. Therefore, the training effectiveness is not affected by a possible CSI reporting periodicity reconfiguration, since the agent decision making is only triggered at each new CSI update. The naive way to calculate the aggregated action composed by the sub-action values chosen in each branch is:

$$\mathbf{a} = (\underset{a_1 \in M}{\operatorname{argmax}} Q_1(s, a_1), ..., \underset{a_K \in M}{\operatorname{argmax}} Q_K(s, a_K)) \quad (19)$$

Using (19) to directly calculate the aggregated action can result in unfeasible actions, i.e. actions that do not satisfy the constraints (5)–(7). If this is the case, a new, feasible, action is recomputed by relying on the action mapping procedure that is going to be described in the next paragraph (line 15).

The Q-function values $Q_k(s, a_k)$ associated to each network branch are computed by extending the aggregation procedure proposed for Dueling Networks. In detail, the same computation reported in (18) is repeated for every branch such that each Q-function value is expressed in terms of the common state-value estimator and the corresponding sub-action advantage. Analytically, the individual Q-function value $Q_k(s, a_k)$ resulting from choosing sub-action $a_k$ in state $s$ is calculated as

$$Q_k(s, a_k) = V(s) + \Big[A_k(s, a_k) - \frac{1}{M} \sum_{a'_k=1}^{M} A_k(s, a'_k)\Big], \quad (20)$$

After each time step, the Q-function values in each branch are updated by means of TD updates with respect to the target value that aggregates the expected future rewards obtainable by each sub-action as

$$y = R(s_t, \mathbf{a}_t) + \frac{\gamma}{K} \sum_{k \in K} Q'_k(s_{t+1}, \underset{a_k \in M}{\operatorname{argmax}} Q_k(s_{t+1}, a_k)), \quad (21)$$

where $Q'_k(s, a_k)$ represents a second DNN initialized with weights $\{\boldsymbol{\theta'} = \boldsymbol{\theta}\}$. Note that the choice of the next sub-action $a_k$ for the time slot $t + 1$ is performed using the current estimated $Q_k(s, a_k)$ but it is then evaluated using the target $Q'_k(s, a_k)$. This procedure, denoted as Double DQN (DDQN), allows to reduce the overestimation problem of the Q-function values that usually affects DQN [32].

The DNN weights $\boldsymbol{\theta}$ of the Q-function are updated in order to approximate the target Q-function values as computed in (21). The total loss is quantified as the expectation of the average TD value across the network branches related to the experience tuples within the mini-batch (lines 17-19), hence it is defined as

$$L(\theta) = \mathbb{E}_{i \in B}\left[\frac{1}{K} \sum_{k=1}^{K} (y_i - Q_k(s, a_k))^2\right], \quad (22)$$

where $y_i$ indicates the target Q-function values obtained by the $i-$th experience tuple. However, differently from the classical experience replay where samples are uniformly drawn from the buffer, we employ the prioritized sampling approach proposed by [33]. The advantage of this procedure is to increase the sampling efficiency by selecting with high probability the experience tuples that are more informative. Practically, experience tuples that provide very different values between the predicted reward and the target reward are the ones the should be sampled more often in order to allow the agent to improve its Q-function estimation. Following this observation, the probability of sampling the $i$th tuple in the experience buffer is:

$$p(i) = \frac{(|\delta(i)| + \xi)^{\alpha_0}}{\sum_j (|\delta(j)| + \xi)^{\alpha_0}}, \quad (23)$$

where $\xi$ is a small positive number that ensures the probability feasibility, $\alpha_0$ tunes the priority of the considered tuple (i.e. the higher $\alpha_0$, the more frequently the tuple is likely to be sampled), and $\delta(i)$ is the cumulative difference between the expected reward $y$ and the predicted Q-function value in each network branch, i.e.

$$\delta(i) = \sum_{k \in K} |y_i - Q_k(s, a_k)|. \quad (24)$$

Finally, since the Q-function computation could suffer from overfitting due to a bias introduced by the sampling probability (23), we weight differently the various losses computed with (22) in order to reduce the magnitude of the gradient updates for the tuples that are sampled more often. In details, each weight $w(i)$ is defined as

$$w(i) = (B \cdot p(i))^{-\beta_0} \quad (25)$$

where $\beta_0$ adjusts the weights importance.

*Action mapping phase.* The proposed agent is not able to determinate whether the select action is feasible. For this reasons, we constrain the action selection procedure during the training phase according to $S_m$ so that the gradient updates are performed only with respect to feasible actions. Whenever the action selected by means of (19) does not satisfy the feasibility constraints, we perform the aggregation of the different branches by solving the following optimization problem, which computes the feasible action maximizing the expected Q-function value in each branch.

Let $Q_k(s, a_k^{(m)})$ be the Q-function value that is estimated by the agent when subchannel $k$ is allocated to MVNO $m$ and $z_{m,k}$ be a binary selection indicator function, which takes value 1 when the $m$-th output neuron of the $k$-th network branch is selected or 0 otherwise. We need to find $z^*_{m,k}$, which maximizes the following

$$\max_z \sum_{k \in K} \sum_{m \in M} z_{m,k} \cdot Q_k(s, a_k^{(m)}) \quad (26)$$

subject to constraints (5)–(7). According to the solution found with (26), the aggregated feasible action at time $t$ is built as

$$\mathbf{a}_t = (\underset{a_1 \in M}{\text{argmax}}\, \widetilde{Q}_1(s, a_1), ..., \underset{a_K \in M}{\text{argmax}}\, \widetilde{Q}_K(s, a_K)), \quad (27)$$

where

$$\widetilde{Q}_k(s, a_k) = \sum_{m \in M} z^*_{m,k} Q_k(s, a_k^{(m)}). \quad (28)$$

This mechanism allows the agent to exclusively learn feasible subchannel allocations, thus improving its convergence speed.

The computation of the optimal solution in (26), can be computationally expensive. This may not be an issue during the training phase, which is performed offline, but makes the algorithm not suitable for the online phase. For this reason, we also propose Algorithm 2, which is a simple low-complexity heuristic algorithm to solve (26) and is suitable for an online execution. The algorithm is based on a greedy scheme that iteratively selects the output neuron providing the highest Q-function value according to $S_m$. Following this procedure, each subchannel is allocated to the MVNO that achieves the highest throughput only if the allocation complies with $S_m$.

**Algorithm 2:** Greedy action mapping procedure

**Input:** $Q_k(s, a_k^{(m)})$, $S_m$
**Output:** Feasible selection indicator function $z^*_{m,k}$

```
1  Initialize z*_{m,k} = 0   ∀m ∈ M, k ∈ K;
2  Initialize subchannel allocation flag
     n_k = 0   ∀k ∈ K;
3  Sort Q_k(s, a_k^(m)) in decreasing order;
4  foreach Q_k(s, a_k^(m)) do
5      if S_m > 0 and n_k = 0 then
6          z*_{m,k} = 1;
7          n_k = 1;
8          S_m = S_m − 1;
9      end
10 end
```

With reference to Algorithm 2, in line 2 we sort in decreasing order all the single $M \cdot K$ Q-function values $Q_k(s, a_k^{(m)})$. For each $Q_k(s, a_k^{(m)})$, in line 3 we check if the assignment policy $S_m$ permits the allocation of subchannels to MVNO $m$ and if subchannel $k$ has not been already allocated. When such condition is met, subchannel $k$ is allocated to MVNO $M$ in line 6 and the related assignment policy and subchannel allocator flag are updated accordingly in line 7 and line 8, respectively. The computation complexity of the described scheme can be computed as follows. The sorting procedure in line 2 has complexity $O(MK \log MK)$. The loop in line 4 iterates all the Q-values, thus the complexity is $O(MK)$. Consequently, the overall asymptotic complexity is $O(MK \log MK)$. Note that, in principle, the discussed greedy scheme could be used to heuristically solve the optimal problem formulation (4) due to the similarity with (26) by using the terms of the summation instead of the Q-function values. However, this is going to provide poor results since the INI depends on the MVNO multiplexing order, that is indeed neglected by the considered scheme. Differently, the proposed DRL approach automatically learns the INI behavior with respect to the subchannel allocation order by means of a trial and error procedure.

### *5.4. Agent deployment in practical scenarios*

The functionality provided by the described agent is performed by the NO, which manages the RAN slicing among the various MVNOs. From a practical perspective, we can fit the agent architecture within the RAN slicing view supported by the well-known Open RAN (ORAN) Alliance. In the context of intelligent RAN virtualization, we can identify the proposed agent as the RAN intelligent controller (RIC) module which is in charge of enhancing the radio resource management [34]. Similarly to the activity performed by our scheme, the RIC leverages artificial intelligence techniques to boost the network capability performance for the sake of a better service provisioning.

## 6. Performance evaluation

We evaluated the performance of the proposed DRL approach by means of simulations. First we provide an overview of the simulation setup, then we discuss the agent scalability performance for different system configurations as well as the effectiveness of the computed allocation policy.

### *6.1. Simulation setup*

We implemented our custom simulator using MATLAB for both the radio network environment and the agent. In order to reliably assess the agent performance, we considered different network scenarios composed by a different number of subchannels and MVNOs. Specifically, we considered multiple RAN bandwidth configurations ranging from 10 MHz to 20 MHz, shared by up to 3 MVNOs of numerologies having subcarrier spacing 15 kHz, 30 kHz, and 60 kHz as dictated by 3GPP specification [25]. Similarly, leveraging the concept of BWP previously introduced, we consider each subchannel having bandwidth $W = 1.5\,\text{MHz}$ that is a value comparable with the RB group granularity size employed in NR physical layer [24]. Since we are interested in quantify the multiplexing gains of different subchannel allocations on the same spectrum, we assume that the BS transmission power is equally allocated across the MVNOs sharing the RAN and that it is scaled according to the number of subchannels. Analogously, we consider that a fixed number of active users, each one located at a random distance $d$ from the BS, is scheduled by the various MVNOs in order to provide the required network

Table 1: Radio parameters

| | |
|---|---|
| Maximum transmission power | 46 dBm |
| Maximum RAN spectrum | 20 MHz |
| Coverage radius | 400 m |
| Carrier frequency | 2.5 GHz |
| Available numerologies | {15, 30, 60} kHz |
| Subchannel bandwidth | 1.5 MHz |
| Number of active users | 12 |
| Small-scale fading statistic | Rayleigh |
| Doppler shift | 35 Hz |
| Fading update | 1 ms |
| Path loss model [35] | $36.7 \log_{10} d + 33.05$ (dB) |
| Shadowing standard deviation | 4 dB |
| Noise PSD | -174 dBm/Hz |

Table 2: Agent parameters

| | |
|---|---|
| Learning rate | $10^{-4}$ |
| Discount factor | 0.3 |
| Exploration decay rate | 0.99 |
| Prioritized sampling | $\alpha_0 = 1$, $\beta_0 = 0.4$ |
| Experience-replay buffer size | $10^5$ |
| Mini-batch size | 32 |
| Episode duration | 100 ms |
| Number of episodes | 500 |

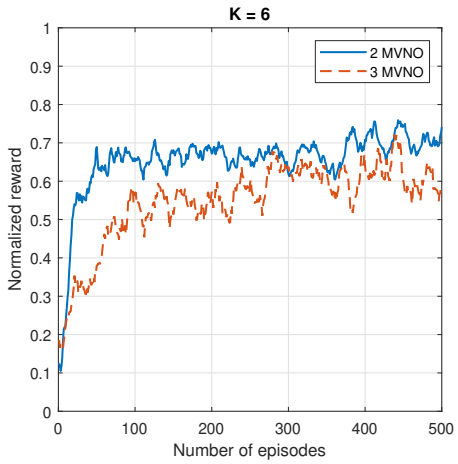
service. The CSI reporting is performed by the users every 1 ms, which also dictates the time step granularity of the episodes during the agent training phase. Moreover, we assume an assignment policy $S_m$ that equally distributes the available subchannels among the MVNOs. When the number of subchannels cannot be equally divided between MVNOs, we sequentially assign the exceeding subchannels starting with the MVNO having the lowest numerology. All the RAN parameters are reported in Table 1.

We selected the agent parameters by experimentally assessing the quality of the obtained results. Specifically, we tuned the DNN number of neurons in each layer as well as the number of hidden layers by incrementally increasing their number in different environment simulations and selecting the combination of values providing the highest agent reward. According to such procedure, the agent DNN has 2 fully connected hidden layers of 1024 and 512 neurons each, which are used to represent the shared network part composed by the common state value estimator module and the sub-actions network branches. Each branch is composed by 1 fully connected hidden layer of 256 neurons. We employ the Rectifier Linear Unit (ReLU) function, $f(x) = \max(0, x)$, as hidden neuron activator. Note that the same hidden layer size is used for all the considered network scenarios, whereas the input and output layers dimension is modified according to the different subchannels and MVNOs configurations. In detail, the input layer, which represents the environment observation, has $K(1 + \sum_m U_m)$ neurons, whereas the aggregated output layer is composed by $K$ branches each one having $M$ neurons. We use stochastic gradient descent to solve (22) and to update the DNN weights $\boldsymbol{\theta}$. Specifically, we trained the agent with mini-batches of 32 samples using the Adam optimizer [36] with learning rate $\alpha = 10^{-4}$ and parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$. The experience tuples are sampled according to the probability distribution generated with prioritization level $\alpha_0 = 1$, whereas the related losses are weighted using a variable $\beta_0$ that is linearly increased from $\beta_0 = 0.4$ to $\beta_0 = 1$ during the first 50 training episodes. Finally, the agent discounts future rewards using a discount rate $\gamma = 0.3$ and it initially explores the environment with probability $\epsilon = 1$ which is then decremented every time step following the update rule $\epsilon \leftarrow \max\{0.1, 0.99\epsilon\}$. In Table 2 we report the overall employed agent parameters.

### *6.2. Scalability performance*

We discuss the agent convergence performance to assess its exploration capabilities for different action space dimensions. In Fig. 4 we plot the normalized reward achieved by the MBRA agent during the training phase for two network configurations composed by 6 and 12 subchannels. For every scenario we considered 2 MVNOs of numerologies 15 kHz and 30 kHz and 3 MVNOs of numerologies 15 kHz, 30 kHz, and 60 kHz are multiplexed on the shared spectrum. The figure show that, regardless of the considered configuration, the agent converges to a stationary reward value, which indicates a successful policy computation. Specifically, we observe that the agent requires more episodes to converge when the number of numerologies and subchannels increases. Such additional training overhead is more visible when the number of subchannels is increased from 6 to 12 rather than an increase of the multiplexed numerologies. Since we assumed independent subchannel fading gains, the agent requires more episodes to infer the relationship between the INI and small-scale fading as the number of subchannel increases. Differently, when we increase the number of numerologies, the correlation between the INI values across contiguous subchannels makes it easier for the agent to exploit its knowledge about the INI dynamic acquired in previous episodes.

In Fig. 5, we show the convergence performance achieved by the agent without the support of the action mapping procedure using the same combination of subchannels and MVNOs previously employed. We plot only the first 250 episodes for the sake of graph visibility. Note that, in this case, we modified the original reward function in order to account for unfeasible subchannel allocations. In this regard, we assigned a null reward to unfeasible actions in order to discourage their choice. We can observe that the agent converges more quickly to a stationary policy that provides a lower expected reward when compared to the agent implementation relying on the action mapping. Intuitively, this behavior is due to the fact that the agent



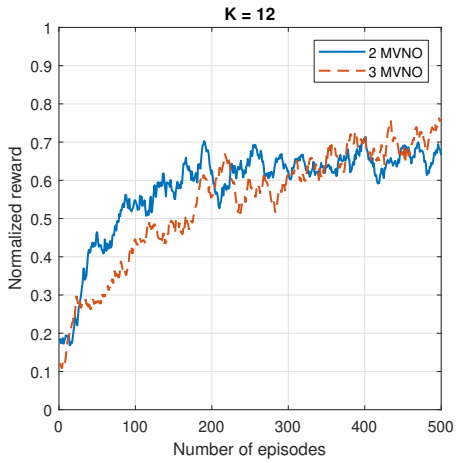
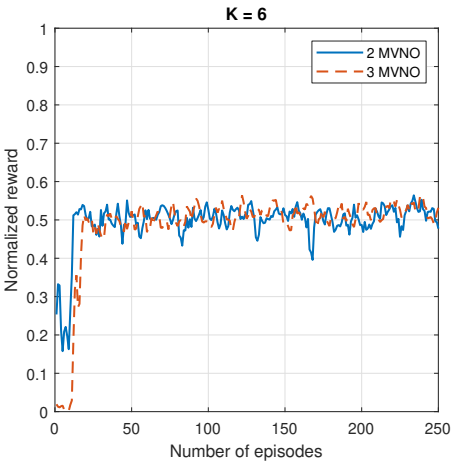




Figure 4: Average reward. Scenarios with 2 MVNOs of numerologies 15 kHz and 30 kHz and 3 MVNOs of numerologies 15 kHz, 30 kHz, and 60 kHz. The number of subchannels is $K = 6$ or $K = 12$.



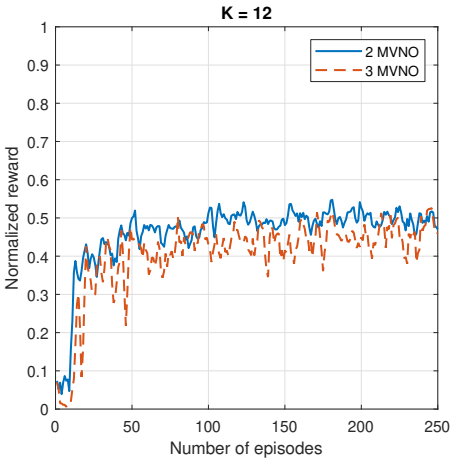




Figure 5: Average reward without the action mapping phase. Scenarios with 2 MVNOs of numerologies 15 kHz and 30 kHz and 3 MVNOs of numerologies 15 kHz, 30 kHz, and 60 kHz. The number of subchannels is $K = 6$ or $K = 12$.

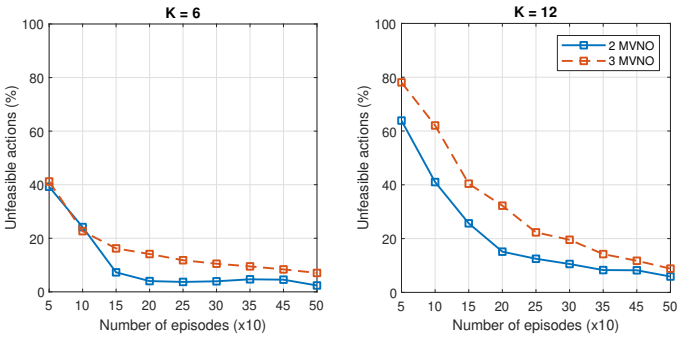



Figure 6: Percentage of unfeasible actions. Scenarios with 2 MVNOs of numerologies 15 kHz and 30 kHz and 3 MVNOs of numerologies 15 kHz, 30 kHz, 60 kHz. The number of subchannels is $K = 6$ or $K = 12$.

learning performance is hindered by the sparsity of the reward function that provides the same low reward value for every unfeasible action. Consequently, as we experimentally observed, the agent learns to how avoid unfeasible actions, which in turn provides an increase of the expected reward, rather than to how select a feasible subchannel allocation providing the highest reward. The effect of this strategy makes the agent converge faster to a poor policy since it is mostly discarding unfeasible actions without actually learning a suitable subchannel allocation.

Fig. 6, shows the agent's capability to gradually learn the feasible action space. Specifically, we plot the percentage of unfeasible actions selected by the agent as a function of the number of episodes. In the figure, the percentage of unfeasible actions decreases with the number of episodes. This behavior is consistent with our observation that the action mapping scheme allows the agent to autonomously select feasible actions as its knowledge about the environment grows. In fact, the agent requires more episodes to discriminate a feasible action space when the number of numerologies and/or subchannel increases. Moreover, the learning procedure is less efficient when 12 subchannels are considered. This is due to the fact that the agent learns the feasible action space according to its own estimate of the expected reward, which is iteratively computed during the learning process. Therefore, estimation errors can affect the action mapping phase. This trend is more evident in complex scenarios that consist of a larger action space. Nonetheless, this analysis highlights the benefits derived from the action mapping phase in term of higher expected reward gains.

### *6.3. Allocation policy performance*

We now assess the performance of the subchannel allocation policy implemented by the trained agent. In addition to the system configuration employed in the previous subsection, we also consider the spectrum allocation for 8 and 10 subchannels in order to provide a more insightful understanding of the policy performance as the system complexity is gradually increased. The results are averaged across 10 independent episodes composed by a different distribution of the users over the BS coverage area. We compare the results obtained by the MBRA agent to the results obtained the the following allocation schemes.

- *Optimal allocation:* the subchannel allocation is computed by performing an exhaustive search of all feasible solutions and selecting the one maximizing (4) at every CSI update. We remark that this subchannel allocation provides the highest aggregated throughput performance so it is used as an upper bound to assess the solution quality of the others schemes.

- *Single branch allocation:* the subchannel allocation is computed using the DRL agent proposed in our previous work [6], which is based on DQN. We will refer to this agent as "single branch resource allocation (SBRA) agent". Differently from the MBRA agent architecture, the SBRA agent employs a single-branch DNN whose output neurons encode every feasible subchannel allocations with a unique index used as identifier. In other words, the action space is a scalar value ranging the number of feasible allocation. Note that, except for the action space structure, such agent employs the same environment and reward definitions as the multi-branch counterpart.

- *INI-aware allocation:* the subchannel allocation is computed by approximating the INI contribution using a model-based approach that is based on the work proposed in [19]. Since the original scheme is used in a different context, we adapted the algorithm to fit our system model in order to provide a fair performance comparison. In detail, the authors defined an INI upper bound, $\sigma^{max}_{INI}$, which is the maximum INI power generated between subbands of different numerologies separated by a guard band of arbitrary size. In the considered scenario, we selected a guard band size depending on the employed numerology. Specifically, the size is computed as $30 \cdot 2^{\mu_m}$ kHz (in other words, the 2 outermost subcarriers in each subchannel are used as guard band). By expressing the subchannel SINR using $\sigma^{max}_{INI}$, i.e.

$$\gamma^u_m(k) = \frac{P_T(k) g^u_m(k)}{\sigma^2_w + g^u_m(k)\sigma^{max}_{INI}}, , \tag{29}$$

  we make (4) convex, hence it can be solved using classical integer linear programming techniques. Note that this alternative formulation underestimates the obtainable data rate since it considers that a fixed INI power is always generated. To assess the optimality gap, we used the Matlab Optimization toolbox to compute the subchannel allocation provided by the simplified SINR expression (29) and then we evaluated the data rate using the exact INI formulation (4).

- *Static allocation:* a random subchannel allocation is drawn among the ones available and it is kept fixed

for the whole episode duration regardless of the CSI and INI values.

We now discuss the per-slice throughput performance. Note that since we are maximizing the cumulative throughput the aggregate data rate of the optimal approach provides, of course, the highest data rate value. However, this is behavior is not guaranteed at the per-slice performance level, hence the various suboptimal schemes might outperform the optimal allocation for a particular MVNO in some scenarios.

In Fig. 7, we plot the throughput achieved by 2 MVNOs sharing the RAN. In general, we observe that the MVNO having 15 kHz numerology has lower data rata due to the asymmetric INI power behavior previously discussed. The MBRA agent provides a good approximation of the optimal solution in all scenarios. In details, we note that the optimality gap increases as more subchannels are multiplexed. This trend is related to the larger action space that the agent has to explore in order to infer the optimal policy. Nonetheless, the proposed agent achieves a higher throughput gain with respect to the SBRA agent that shows a more visible performance loss when 10 and 12 subchannels are available. The large dimensionality of the action space hampers the SBRA agent, since it cannot leverage the flexibility provided by the multi-branch architecture to efficiently explore all the possible subchannels combinations. Moreover, the INI-aware algorithm is outperformed by the DRL based schemes since its INI power overestimation limits the subchannel multiplexing gain as instead occurs when the exact INI power computation is employed. Similarly, the static allocation provides the lowest performance since it does not account neither for the wireless channel condition nor for the INI power.

In Fig. 8, we repeat the analysis when 3 MVNOs are active. Generally, we observe a degradation of the DRL based schemes due to the increased action space complexity that is characterized by subchannel allocations of three different numerologies. In detail, when 6 and 8 subchannels are available, the MBRA and SBRA agents achieve comparable results. In this scenario, we observe that the agents adopt different allocation strategies. Unlike the fair allocation of the SRBA agent, the MBRA ensures a higher throughput to the 30 kHz MVNO at the expense of the 15 kHz MVNO, which is served with a lower data rata. This fact provides some intuitive insight about the problem complexity since it shows that multiple allocation may provide similar performance. For the cases of 10 and 12 subchannels, the gap of the INI-aware approach from the optimal scheme is smaller than the previous case since the higher number of numerologies increases the INI power and makes the INI power overestimation much more reliable. In particular, this effect is highlighted by 15 KHz MVNO performance. As a matter of fact, by increasing the number of subchannels assigned to the 30 kHz and 60 kHz MVNOs, the subcannel allocation combinations that generate INI on the 15 kHz MVNO are also larger. Consequently, it is more challenging for the DRL schemes to find a solution that simultaneously ensures the same performance of the 10 subchannel case for all three slices. Lastly, the static allocation is outperformed by all the schemes.

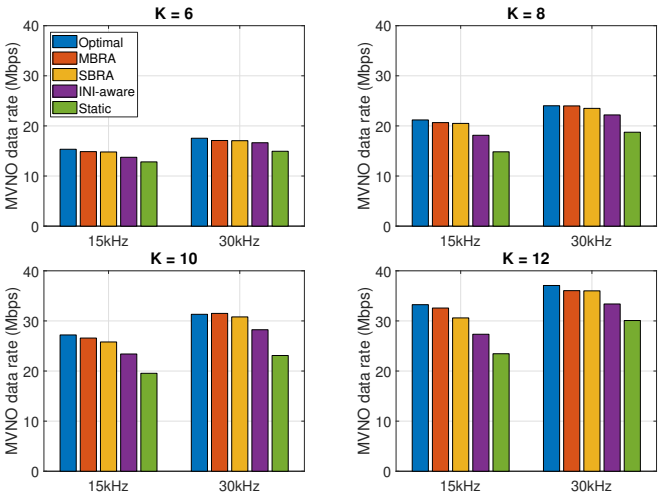



Figure 7: Aggregated MVNO throughput. Scenario with 2 MVNOs of numerologies 15 kHz and 30 kHz. The number of subchannels ranges from $K = 6$ to $K = 12$.

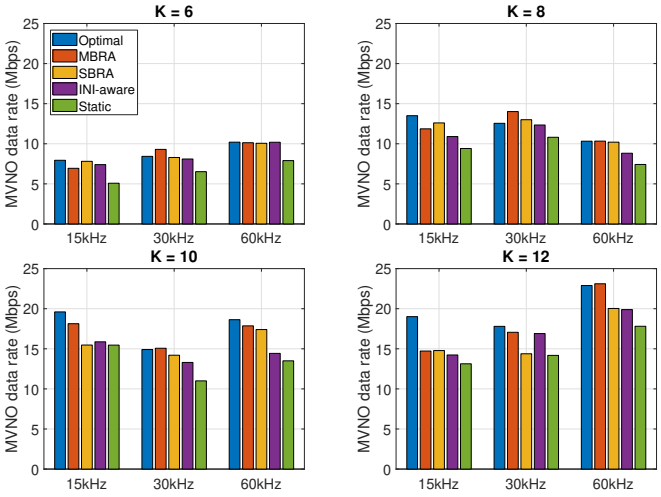



Figure 8: Aggregated MVNO throughput. Scenario with 3 MVNOs of numerologies 15 kHz, 30 kHz, and 60 kHz. The number of subchannels ranges from $K = 6$ to $K = 12$.

In Fig. 9, we provide a deeper analysis of the agent performance for different slicing assignment policy. We consider the scenario of 2 MVNOs and 12 subchannels and we plot the subchannel data rate as a function of the number of subchannels assigned by $S_m$ to each MVNO. In general, we note that as more subchannels are assigned to the same MVNO, the per-slice subchannel data rate increases since less INI is generated by the other MVNO that has access to a fewer number of spectrum resources. This trend also impacts the optimality gap of all schemes that gets smaller as the MVNO having the majority of the resources approaches a subchannel allocation comparable to a single numerology scenario which is INI free. The

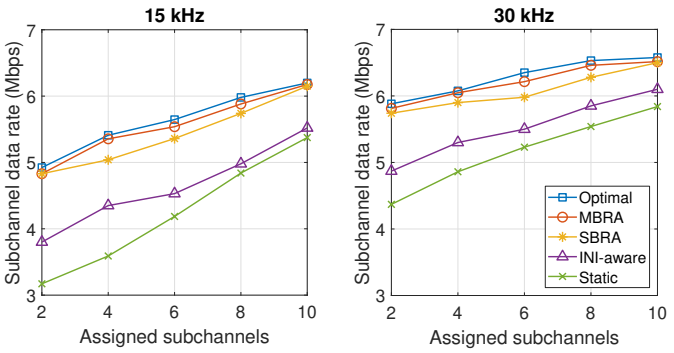



Figure 9: MVNO subchannel data rate for different subchannel assignment policies. Scenario with 2 MVNOs of numerologies 15 kHz and 30 kHz. The number of subchannels is $K = 12$.

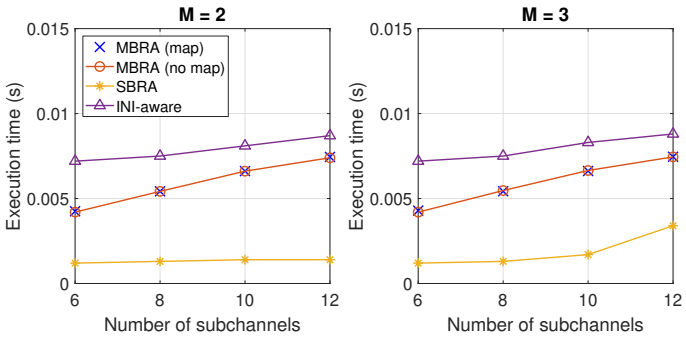



Figure 10: Execution time required for the subchannel allocation computation. Scenario with 2 MVNOs of numerologies 15 kHz and 30 kHz and 3 MVNOs of numerologies 15 kHz, 30 kHz, and 60 kHz. The number of subchannels ranges from $K = 6$ to $K = 12$.

MBRA scheme provides the best performance followed by the SBRA agent that has a wider gap from the optimal allocation when the subchannels are equally distributed between the two MVNO. This behavior derives from the fact that this assignment policy configuration provides the highest number of feasible subchannel allocations that hinders the SBRA agent convergence performance.

Finally, in Fig. 10, we discuss the computational complexity of the MBRA, SBRA and INI-aware schemes in calculating the subchannel allocation at each CSI update as the number of subchannel increases when 2 and 3 MVNOs are active. We neglected the optimal and static allocations since the former relies on an exhaustive search approach that it is not meant to be used as a practical solution, whereas the latter does not involve any actual subchannel allocation as it is fixed at the beginning of each episode. In general, we observe that the execution time is more sensitive to the increase of the number of subchannel compared to the number of slices. All schemes are comparable as it shown by the similar execution times. Nonetheless, the DRL based schemes provide the best performance with the SBRA achieving the lowest computational time. This is an expected behavior since the DNN size of the MBRA agent is higher than the SBRA agent due to the fact that additional network branches are added to the neural network architecture for each new subchannel. Consequently, the DNN feed-forward procedure is computationally more expensive as more neurons are involved in the computation. Moreover, we note the negligible impact of the action mapping procedure on the overall MBRA agent performance as it is indicated by the overlapping curves showing the execution times of the agent when the action mapping module is employed due to the action unfeasibility and when it is not employed. Despite the lower SBRA computational complexity for all the considered network configurations, it is important to highlight the considerable increases of the execution time of this scheme when the number of subchannels is 12 and 3 MVNOs are active. In this scenario, the high number of available actions makes the computation of the maximum value of the Q-function costly since the agent has to assess the value of each output neuron. Differently, the MBRA agent provides a more contained computational time overhead that shows the benefit of the proposed architecture in term of scalability. However, we remark that SBRA agent is a valid option for system scenarios of limited complexity since it achieves results that are comparable to the ones provided by the MBRA agent exploiting a simpler and computationally lighter agent implementation.

## 7. Conclusion

We study the problem of maximizing the aggregated throughput performance of users belonging to different MVNOs in presence the INI generated from the simultaneous multiplexing of spectrum slices having heterogeneous numerologies. To avoid the computational complexity of directly solving the optimization problem, we leverage the DRL theory to train an agent capable of finding a sub-optimal solution. Our proposed agent exploits the uncorrelated small-scale fading fluctuations to mitigate the INI and, at the same, to increase the data rate of the users. The scalability of the solution is obtained by means of a multi-branch agent architecture that individually considers the allocation of every subchannel to one of the active MVNOs. In addition, we enhance the convergence performance by proposing an action mapping procedure to guarantee the action feasibility. We evaluate the performance of the proposed agent versus several allocation schemes. Results show that the multi-branch agent provides a spectrum allocation policy comparable with optimal solution in most of the considered scenarios.

## References

[1] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. De Silva, F. Tufvesson, A. Benjebbour, G. Wunder, 5G: A tutorial overview of standards, trials, challenges, deployment, and practice, IEEE journal on selected areas in communications 35 (6) (2017) 1201–1221.

[2] A. B. Kihero, M. S. J. Solaija, A. Yazar, H. Arslan, Inter-numerology interference analysis for 5G and beyond, in: 2018 IEEE Globecom Workshops (GC Wkshps), IEEE, 2018, pp. 1–6.

[3] X. Zhang, L. Zhang, P. Xiao, D. Ma, J. Wei, Y. Xin, Mixed numerologies interference analysis and inter-numerology interference cancellation for windowed OFDM systems, IEEE Transactions on Vehicular Technology 67 (8) (2018) 7047–7061.
[4] R. Li, Z. Zhao, Q. Sun, I. Chih-Lin, C. Yang, X. Chen, M. Zhao, H. Zhang, Deep reinforcement learning for resource management in network slicing, IEEE Access 6 (2018) 74429–74441.
[5] L. Liang, H. Ye, G. Y. Li, Spectrum sharing in vehicular networks based on multi-agent reinforcement learning, IEEE Journal on Selected Areas in Communications 37 (10) (2019) 2282–2292.
[6] M. Zambianco, G. Verticale, Spectrum allocation for network slices with inter-numerology interference using deep reinforcement learning, in: 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications, IEEE, 2020, pp. 1–7.
[7] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, L.-C. Wang, Deep reinforcement learning for mobile 5g and beyond: Fundamentals, applications, and challenges, IEEE Vehicular Technology Magazine 14 (2) (2019) 44–52.
[8] L. Liang, H. Ye, G. Yu, G. Y. Li, Deep-learning-based wireless resource allocation with application to vehicular networks, Proceedings of the IEEE (2019).
[9] H. Ye, G. Y. Li, Deep reinforcement learning for resource allocation in v2v communications, in: 2018 IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–6.
[10] S. E. Elayoubi, S. B. Jemaa, Z. Altman, A. Galindo-Serrano, 5G RAN slicing for verticals: Enablers and challenges, IEEE Communications Magazine 57 (1) (2019) 28–34.
[11] A. A. Zaidi, R. Baldemair, V. Molés-Cases, N. He, K. Werner, A. Cedergren, OFDM numerology design for 5G new radio to support IoT, eMBB, and MBSFN, IEEE Communications Standards Magazine 2 (2) (2018) 78–83.
[12] Y. Shi, Y. E. Sagduyu, T. Erpek, Reinforcement learning for dynamic resource optimization in 5g radio access network slicing, in: 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), IEEE, 2020, pp. 1–6.
[13] B. Khodapanah, A. Awada, I. Viering, A. N. Barreto, M. Simsek, G. Fettweis, Slice management in radio access network via deep reinforcement learning, in: 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), IEEE, 2020, pp. 1–6.
[14] A. Anand, G. De Veciana, S. Shakkottai, Joint scheduling of URLLC and eMBB traffic in 5G wireless networks, IEEE/ACM Transactions on Networking (2020).
[15] J. Zhang, X. Xu, K. Zhang, B. Zhang, X. Tao, P. Zhang, Machine learning based flexible transmission time interval scheduling for eMBB and URLLC coexistence scenario, IEEE Access 7 (2019) 65811–65820.
[16] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, H. Mineno, Flexible resource block allocation to multiple slices for radio access network slicing using deep reinforcement learning, IEEE Access 8 (2020) 68183–68198.
[17] A. F. Demir, H. Arslan, Inter-numerology interference management with adaptive guards: A cross-layer approach, IEEE Access 8 (2020) 30378–30386.
[18] A. Yazar, H. Arslan, Reliability enhancement in multi-numerology-based 5G new radio using ini-aware scheduling, EURASIP Journal on Wireless Communications and Networking 2019 (1) (2019) 110.
[19] L. Marijanovic, S. Schwarz, M. Rupp, Multi-user resource allocation for low latency communications based on mixed numerology, in: 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), IEEE, 2019, pp. 1–7.
[20] J. Choi, B. Kim, K. Lee, D. Hong, A transceiver design for spectrum sharing in mixed numerology environments, IEEE Transactions on Wireless Communications 18 (5) (2019) 2707–2721.
[21] K. S. Chandran, C. Ali, Filtered-ofdm with index modulation for mixed numerology transmissions, in: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), IEEE, 2020, pp. 306–310.
[22] E. Memisoglu, A. B. Kihero, E. Basar, H. Arslan, Guard band reduction for 5g and beyond multiple numerologies, IEEE Communications Letters 24 (3) (2019) 644–647.
[23] Y. Varun, K. S. Chandran, C. Ali, Inter-numerology interference reduction based on precoding for multi-numerology ofdm systems, in: 2020 IEEE 3rd 5G World Forum (5GWF), IEEE, 2020, pp. 542–546.
[24] 3GPP, NR; Requirements for support of radio resource management, Technical Specification (TS) 38.133, 3rd Generation Partnership Project (3GPP), version 15.3.0 (2018).
[25] 3GPP, NR; Physical channels and modulation, Technical Specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), version 15.5.0 (2019).
[26] R. Horst, H. Tuy, Global Optimization: Deterministic Approaches, Springer Science & Business Media, 2013.
[27] R. S. Sutton, A. G. Barto, et al., Introduction to reinforcement learning, Vol. 135, MIT press Cambridge, 1998.
[28] C. J. Watkins, P. Dayan, Q-learning, Machine learning 8 (3-4) (1992) 279–292.
[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602 (2013).
[30] A. Tavakoli, F. Pardo, P. Kormushev, Action branching architectures for deep reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.
[31] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, N. Freitas, Dueling network architectures for deep reinforcement learning, in: International conference on machine learning, 2016, pp. 1995–2003.
[32] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 30, 2016.
[33] T. Schaul, J. Quan, I. Antonoglou, D. Silver, Prioritized experience replay, arXiv preprint arXiv:1511.05952 (2015).
[34] S. Niknam, A. Roy, H. S. Dhillon, S. Singh, R. Banerji, J. H. Reed, N. Saxena, S. Yoon, Intelligent o-ran for beyond 5g and 6g wireless networks, arXiv preprint arXiv:2005.08374 (2020).
[35] ITU-R, Guidelines for evaluation of radio interface technologies for IMT-2020, Tech. rep., International Telecommunication Union (ITU) (2017).
[36] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).