Hierarchical Beamforming in Random Access Channels

Adrian Agustin
Advanced Signal and Information Processing (ASIP)
Centre Tecn de Telecom. de Catalunya (CTTC/iCERCA)
adrian.agustin@cttc.cat

Josep Vidal, Margarita Cabrera-Bean
Signal Processing and Communications (SPCOM)
Universitat Politècnica de Catalunya (UPC)
{josep.vidal, marga.cabrera}@upc.edu

***Abstract*—Managing a massive number of terminals in a contention-based multiple access is challenging due to its intrinsic limited efficiency. For example, in the random access channel considered in LTE-A and 5G NR, Base Station (BS) is just aware of the collided and non-collided preambles. Several time-based protocols have been investigated to redistribute the overload under high terminal activity, thus avoiding the congestion. In this work, we explore the use of the spatial domain by means of a hierarchical codebook-based beamforming, where the BS selects the appropriate beams as a function of the number of non-collided and collided preambles. Since the activity and placement of terminals may be dynamic over time, the sequential selection of parameters can benefit from a reinforcement learning (RL) framework. We propose an algorithm that can exploit both domains, temporal and spatial, with the goal of reducing collisions and enhancing transmission delay. Our approach is able to efficiently learn whenever there is a non-homogeneous spatial distribution of terminals and adapt the spatial beams accordingly.**

***Keywords*—Hierarchical Beamforming, Deep Reinforcement Learning, Random Access Channel**

I. Introduction

In recent years we have witnessed the rapid introduction of a diverse Internet of Things (IoT) devices communicating wirelessly in scenarios like smart cities or industry 4.0. This number is expected to increase significantly in the near future so that wireless communication systems will be confronted to deal with a massive number of terminals, [1]. In a contention-based multiple access scenario, the randomness in the generation of communication requests, the huge number of terminals and the limited spectrum are the causes of congestion. This drawback already comes up under resource reservation-based protocols (like in LTE-A and 5G NR), because there is an initial requesting phase (i.e. Random Access Channel (RACH)) where terminals contend on the same resources.

The ACB (Access Class Barring) mechanism has been extensively investigated in the past with the objective of regulating the access of competing terminals in the RACH, see for example [2][3]. In LTE-A and 5G NR networks, the Base Station (BS) broadcasts certain system parameters that terminals must use upon performing a random access (RA) request. One of the parameters is the ACB factor or barring rate. When willing to access, each terminal generates a uniformly distributed random number and, if it is above the ACB factor, then it can proceed with the rest of the phases of the RACH. Otherwise, the terminal is backlogged for a random period of time, $T_{barring}$, until the next allowed random access opportunity (RAO). With the objective of reducing collisions and improving the access efficiency, several schemes have been proposed to optimize the ACB factor based on RL [4]. Taking into account the partial observations of environment, RL-based approaches have the ability to continuously optimize the parameters under changing conditions. Examples on the ACB optimization can be found in [2][3][5][6]. Applications of RL for other optimizations considered in the RACH are in [7].

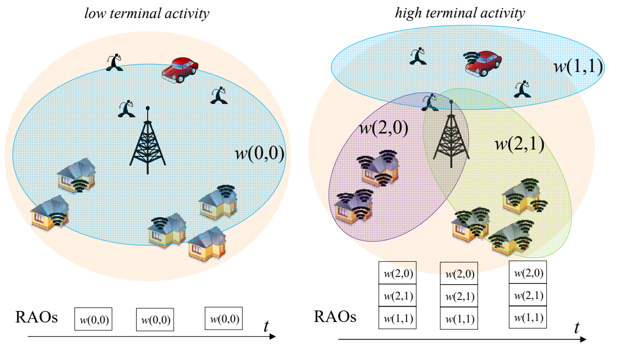

Fig. 1. Beam selection at BS to provide coverage and controlling the collisions as a function of terminal activity. (Left) low terminal activity, (right) high terminal activity. The beam $w(0,0)$ employed on each RAO under low terminal activity, is decomposed into three narrow beams $w(2,0)$, $w(2,1)$, $w(1,1)$ under high terminal activity conditions.

The ACB mechanism focuses on deferring the access of terminals with $T_{barring}$ when the overall terminal activity increases. This comes with the penalty of increasing the delay experienced by terminals when the ACB factor is small, because the barring time is much larger than the backoff time for generating a new RA attempt ($T_{barring}>>T_{BO}$). In this work, we investigate the benefits of exploiting the spatial domain when BS is equipped with multiple antennas and terminals follow a non-homogeneous spatial distribution. We aim to reduce the number of contending terminals by using spatially orthogonal beams along ACB mechanism per beam, thus keeping larger ACB factors and controlling the experienced delay. This is challenging in the RACH since the BS has not any knowledge about the current number and activity of contending terminals and must guarantee that all terminals have the opportunity to transmit. Furthermore, activating multiple parallel beams increases the power consumption of BS, so we would like to employ them only when it will be necessary. Fig 1 illustrates an example of desired spatially coverage area that depends on the multiple input multiple output (MIMO) receiver beam configuration as a function of

The work of A.Agustin was supported by the Spanish Government through the Statistical Learning and Inference for Large Dimensional Communication Systems (ARISTIDES, RTI2018-099722-B-100) Project. The work of J.Vidal and M.Cabrera-Bean was supported by the project ROUTE56 (Agencia Estatal de Investigación, PID2019-104945GB-I00/AEI/10.13039/501100011033), and in part by the Grant 2017 SGR 578 (AGAUR, Generalitat de Catalunya).

the instantaneous terminal activity. Fig 1-left shows that under low terminal activity, i.e. low number of collisions, the BS employs an omnidirectional beam pattern $w(0,0)$ to broadcast system parameters and to detect the RA requests. If instead there are many terminals (see buildings in the lower side of the plot) transmitting new RA requests, the number of collisions increases if the same beam pattern as in Fig.1-left is maintained. The use of multiple beams along with broadcasting specific information per beam, allow terminals to select its preferred RAO, and to reduce the number of collisions, thus reducing average delay and jitter (see Fig.1-right).

A potential solution is providing a hierarchical coverage using a hierarchical codebook design [8]. This solution has been considered for 5G NR and millimeter-wave based communication for beam-alignment purposes [9]. However, in our setup, the selection of codewords is inferred from the number of collided and non-collided access tries.

The present work addresses the reduction of collisions and delays in the random access channel by proposing a RL-based algorithm based on Deep Q Networks [10]. The proposed algorithm consists of two DRL agents that select the suitable beams from a codebook and the ACB factor on each beam, respectively. Reward functions of the agents are defined to take into account the delay of terminals and the number of employed beams, penalizing the use of multiple beams when the terminal activity is low.

II. System Model

A. *Random Access Channel in 3GPP*

The random access procedure in 3GPP [11] consists of exchanging 4 messages. The information required by terminals to use the RACH is broadcasted by the BS in the System Information Blocks (SIB), in particular the one named SIB2. Once a terminal has an access request, it randomly selects one preamble out of M and transmits in the next available random access opportunity time slot (RAO). This is known as message 1 (msg-1) transmission. The BS is listening the RAO and detects which preambles were active. However, since terminals, randomly select the preambles there might happen that a preamble be selected by more than one terminal. During a random access response phase, the BS reports about the preambles correctly detected, without being aware of which ones correspond to a single machine or to more than one, and the reserved resources for those terminals (msg-2). Afterwards, terminals use the assigned resources (msg-3) to communicate with the BS, which acknowledges the message in msg-4. The RA attempt fails in case a preamble has been chosen by more than one terminal, which causes a collision in msg-3. If a terminal does not receive the msg-4, it tries a new access attempt after a random backoff time defined by,

$$T_{BO} = U\left(0, BI\right), \quad BI = 0.960 \text{ s} \tag{1}$$

where $U(0,A)$ denotes a random number generated using a uniform distribution in $[0,A)$, and BI is a parameter transmitted in the SIB. Furthermore, if a terminal has attempted more than 10 times, a failure is declared. Our objective is to improve the contention (i.e. msg-1 and msg-2) in the selection of the preamble.

The ACB mechanism allows redistributing an overload of access requests over time. When a terminal has an access request, it generates a uniformly distributed random number p between 0 and 1. If $p \leq P_{ACB}$ then the terminal selects a preamble and starts the RACH procedure described previously. Otherwise, if $p > P_{ACB}$ then the terminal must wait for the next attempt a period of time defined by,

$$T_{barring} = \left(0.7 + 0.6 \times U\left(0,1\right)\right) \times T_{ACB}, \quad T_{ACB} = 4 \quad \text{s} \tag{2}$$

The BS periodically broadcasts the mean barring times, T_{ACB} and ACB factor, P_{ACB} once every T_{SIB2}. In this work if a terminal has attempted more than 10 times the ACB mechanism, then we declare a failure. Notice that $T_{barring}$ is much larger than T_{BO}, thus using $P_{ACB}<1$ increases the delay.

B. *Hierarchical Codebook design*

Hierarchical codebook design has been considered in millimeter-wave communications to provide an efficient search over the large number of candidate beam directions, see for example [8]. Basically, it provides a structure of hierarchy of codewords with different spatial resolution as it is depicted in Fig 2. Each codeword is associated to a beam with a given beamwidth. The first layer l=1 defines a single codeword $w(0,0)$ for omnidirectional reception. This can be improved by the l+1 layer with the codewords $w(1,0)$ and $w(1,1)$ and so on with the ensuing layers, each one covering half of the space covered by their preceding codeword in the l-1-th layer .

On each RAO a collection of beams are selected to provide coverage to the whole space. In case we identify a large number of collisions in a codeword $w(l,n)$, we might decompose it into two codewords with smaller beamwidths, so that the number of terminals in each beam is reduced,

$$w\left(l,n\right) \to \begin{cases} w\left(l+1,k\right) \\ w\left(l+1,k+1\right) \end{cases}, \quad k = 2\left\lfloor \frac{n+1}{2} \right\rfloor \tag{3}$$

In contrast, in case of a small number of collisions in two codewords, we might consider the possibility of merging their beams by employing codewords of broader beamwidth.

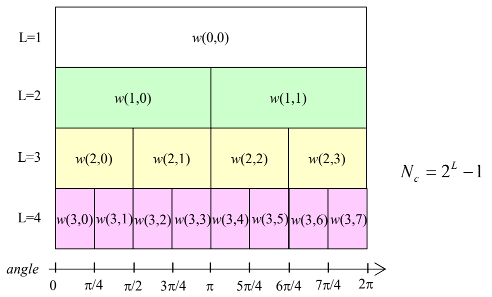

Fig. 2. Structure of the beam coverage and beam codeword as a function of number of layers. There are up to N_c different codewords.

C. *Terminal distribution*

Our scenario is a heterogeneous machine-type communications one, with terminal distribution following the guidelines described in [12]. Two types of terminals are assumed during a period of 20 seconds, both using a limited Beta(3,4) profile: high and low priority terminals. The first ones generate random requests starting at time t=5 s and they are placed in a spatial domain sector of $[0,\pi]$. The low priority terminals start at t=7s and are distributed uniformly over the entire space domain $[0,2\pi]$.

D. Time resources

Fig. 3 shows how the time resources are grouped. Every T_{FRAME} seconds, the BS selects the pattern of beams to apply in all RAOs, ensuring that all angular space is covered. In the first frame (n_{frame}=0) BS uses omnidirectional codeword $w(0,0)$. The BS decides to apply the pattern of codewords $w(2,1)$, $w(2,2)$, $w(1,1)$ (n_{frame}=6) in the second frame. Finally in the third frame, BS applies the pattern $w(1,0)$, $w(1,1)$ (n_{frame}=1).

Terminals are aware of changes in the system after decoding the SIB2 message, transmitted every T_{SIB2} seconds. Then they have information about ACB factor, i.e. P_{ACB}, and the structure of the frame. This information is transmitted using the beam structure, so each terminal knows the RAO (and its associated beam) when it has to transmit just by comparing the received power. Every T_{SIB2}, the BS decides the ACB factor to be applied on the associated beam and type of terminal. How these parameters (P_{ACB} and frame structure) are selected is addressed in the next section.

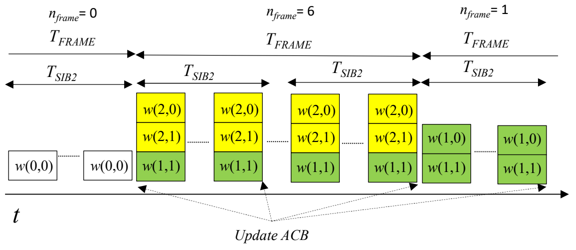

Fig. 3. Temporal RAO configuration. Codewords of the same color belong to the same layer.

III. Reinforcement Learning architecture

The proposed architecture is based on two independent RL-agents working at different time scales for controlling the ACB factor (DQN-ACB) and the frame structure (DQN-FRAME), respectively. Both are based in Deep Q-Networks [10] with the objective of dealing with correlated inputs and outputs and the non-stationarity of the process that conventional Q-learning is not able to capture. In particular, conventional Q-learning minimizes,

$$L_Q(\theta) = \mathrm{E}\left[\left(r_t + \gamma \max_{a'} Q_\theta(s',a') - Q_\theta(s,a)\right)^2\right] \tag{4}$$

where r_t is the reward obtained when the agent is at state s and performs action a, Q is the state-action value function, θ denotes the parameters of the neural network (NN) that predicts function Q, γ is the discount factor that takes into account the impact of future rewards, and a, a' are the RL-agent actions at current state s, and future state s', respectively.

DQN [10] improves the previous architecture by introducing an experience replay memory (ERM), where transitions are stored, and using two NNs Q approximators, named target network $Q_{\theta'}(s',a')$ and local network $Q_\theta(s,a)$ with the same structure and parameters θ' and θ, respectively. The stored experiences are employed (randomly sampling) to form minibatches using the target network, $Q_{\theta'}$. Afterwards, the local network, Q_θ, is trained, which is responsible of computing the expectation of the long-term reward. In order to improve the stability of the optimization, the parameters of the target network $Q_{\theta'}$ are updated infrequently with the learned weights of the local network, θ. We provide a brief description of the DQN algorithm in (6). The function to be minimized in DQN is,

$$L_{DQN}(\theta,\theta') = \mathrm{E}\left[\left(r_t + \gamma \max_{a'} Q_{\theta'}(s',a') - Q_\theta(s,a)\right)^2\right] \tag{5}$$

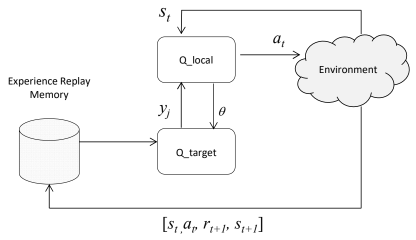

Fig. 4. Deep Q-Networkd proposed in [10]

We apply the RL-based framework for designing an online optimization algorithm where our actions are the ACB factor, P_{ACB}, and the frame structure (pattern of codewords in a frame) selection. Nevertheless, according to Fig.3 in section II.D, these actions are performed at different time scale (T_{SIB2}, T_{FRAME}). In this regard we propose the use of two parallel DQNs agents as it is depicted in Fig. 5. We assume the action selected by the DQN-FRAME, responsible of the frame structure, is an input to the state of the DQN-ACB which will select the ACB factor (dotted lines in Fig. 5).

Algorithm DQN

```
Initialize networks $Q_\theta, Q_{\theta'}$
Initialize experience replay memory (ERM)
for t=1,T do
    With probability $\varepsilon$ select a random action $a_t$
    otherwise select $a_t = \max_a Q_\theta(s_t,a)$
    Perform the action and store $(s_t,a_t,r_{t+1},s_{t+1})$ in ERM                (6)
    Sample a random minibatch of transitions from ERM ($N_s$)
        $y_j = r_t + \max_{a'} Q_{\theta'}(s_{t+1},a')$
    Train $Q_\theta$ local network: $L_{DQN}(\theta) = \frac{1}{N_s}\sum_j \left(y_j - Q_\theta(s_t,a_t)\right)^2$
    if update target network
        $\theta' = (1-\tau)\theta' + \tau\theta$
```

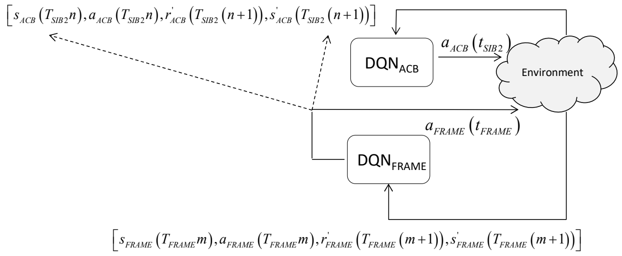
$[s_{ACB}(T_{SIB2}n), a_{ACB}(T_{SIB2}n), r'_{ACB}(T_{SIB2}(n+1)), s'_{ACB}(T_{SIB2}(n+1))]$
DQN$_{ACB}$
$a_{ACB}(t_{SIB2})$
Environment
$a_{FRAME}(t_{FRAME})$
DQN$_{FRAME}$
$[s_{FRAME}(T_{FRAME}m), a_{FRAME}(T_{FRAME}m), r'_{FRAME}(T_{FRAME}(m+1)), s'_{FRAME}(T_{FRAME}(m+1))]$

Fig. 5. Proposed structure of parallel DQN, one for optimizing the ACB factor and other for the frame structure, where n,m represent the different time scale. The state s_{ACB} takes into account a_{FRAME}

A. DQN for ACB selection

The DQN-ACB agent will try to find the optimal value of P_{ACB} that allows controlling the number of simultaneous random access requests and the maximum transmission delay. We describe below the main parameters of this network.

1) Reward function

We would like that the ACB mechanism controls the number of terminals that can start the random access request, but at the same time, have some information about the delay of succeeding terminals. To this end, we propose the following reward function for the terminals of type k, for k=0,1, those are served in RAOs using the c-th codeword,

$$f_{c,ACB}^{k}\left(nT_{SIB2}\right)=\frac{N_{d,c}^{k}}{\left(1+\delta_{\max,c}^{k}\right)} \tag{7}$$

with $N_{d,c}^{k}$ the number of successfully detected terminals of type k and $\delta_{\max,c}^{k}$ the maximum delay under the c-th codeword during the last T_{SIB2} period. The total reward is defined as a weighted combination of rewards obtained by the both types of terminals,

$$r_{c}^{ACB}\left(nT_{SIB2}\right)=\mu_0 f_{c,ACB}^{0}\left(nT_{SIB2}\right)+\mu_1 f_{c,ACB}^{1}\left(nT_{SIB2}\right) \tag{8}$$

where μ_0, μ_1 are the weights to adjust the priority of the different types of terminals.

2) Action set

Each type of terminals has its own ACB factor, which is based on a discrete array of 8 values A_s=[0.1 0.2 0.4 0.5 0.6 0.7 0.8 1]. Since we are considering two types of terminals, the action selected by the DQN corresponds to a joint selection of ACB factor,

$$\left\{P_{c,ACB}^{0}\left(nT_{SIB2}\right),P_{c,ACB}^{1}\left(nT_{SIB2}\right)\right\}=\phi\left(a_t^{c}\left(nT_{SIB2}\right)\right) \tag{9}$$

with $\phi()$ denoting the mapping between the action and the ACB factors to be selected. For example a_t=0 denotes $P_{ACB}^{0}=P_{ACB}^{1}=0.1$. Notice that the total number of actions to be considered is 8×8 = 64

3) State

The state for the c-th codeword at T_{SIB2} is defined as,

$$\left[N_{d,c}^{0},P_{c,ACB}^{0},\delta_{\max,c}^{0},N_{coll,c},N_{d,c}^{1},P_{c,ACB}^{1},\delta_{\max,c}^{1},n_{frame},c\right] \tag{10}$$

with c the employed codeword, n_{frame} the frame selected on the DQN running every T_{FRAME}, $\delta_{\max,c}^{k}$ is the maximum measured delay, and $N_{coll,c}$ the number of collisions at the RAOs using the c-th codeword during the last T_{SIB2}.

B. DQN for frame selection

The DQN-FRAME agent will try to find the optimal pattern of codewords in order to provide coverage to the whole space, see Fig.3. We describe in the following the main parameters of this network.

1) Reward

First, we define the following parameters,

$$\begin{aligned}\tilde{N}_{d,c}\left(mT_{FRAME}\right)&=\sum_{n=(m-1)\omega}^{m\omega-1}N_{d,c}^{0}\left(nT_{SIB2}\right)+N_{d,c}^{1}\left(nT_{SIB2}\right)\\ \tilde{N}_{coll,c}\left(mT_{FRAME}\right)&=\sum_{n=(m-1)\omega}^{m\omega-1}N_{coll,c}\left(nT_{SIB2}\right),\quad \omega=\frac{T_{FRAME}}{T_{SIB2}}\end{aligned} \tag{11}$$

where $\tilde{N}_{coll,c}$,$\tilde{N}_{d,c}$ are the number of collided and non-collided preambles over the RAOs using the c-th codeword over the different T_{SIB2} present in a given T_{FRAME}. Likewise, we want to take into account the maximum delay experienced by high priority terminals (k=0),

$$\begin{aligned}\tilde{\delta}_{\max,c}\left(mT_{FRAME}\right)&=\max_{m\in[(m-1)\omega,\dots,m\omega-1]}\delta_{\max,c}^{0}\left(nT_{SIB2}\right)\\ \tilde{\delta}_{\max}\left(mT_{FRAME}\right)&=\max_{c\in\psi\left(n_{frame}\right)}\tilde{\delta}_{\max,c}\left(nT_{SIB2}\right)\end{aligned} \tag{12}$$

where $\psi(n_{frame})$ is the set of codewords associated to the current frame acction, n_{frame} which will be defined in (14), accounts for the codeword structure, mT_{FRAME} is the time instants, and ω is the ratio defined in (11).

The total reward for a given frame configuration is given by combining all rewards obtained on the different active codewords and the worst experienced delay. Likewise, to avoid that BS will always transmit with all beams, so the power consumption increases, the obtained reward will be penalized with the number of actives beams,

$$r^{FRAME}\left(mT_{FRAME}\right)=\frac{1}{\kappa}\frac{\tilde{N}_{d,c}\left(mT_{FRAME}\right)}{1+\tilde{\delta}_{\max}\left(mT_{FRAME}\right)} \tag{13}$$

where κ is the number of active beams per RAO.

2) Action set

We define up to 21 frame structures using combinations of codewords obtained from a codebook of L=4. The action selected, n_{frame}, establishes the pattern of codewords. In Fig.3, n_{frame} =6 and n_{frame} =1 were considered, with κ=3 and κ=2 beams, respectively.

$$\psi\left(n_{frame}\right)=\begin{cases}0: w_{(0,0)}\\ 1: w_{(1,0)},w_{(1,1)}\\ 2: w_{(1,0)},w_{(2,2)},w_{(2,3)}\\ 3: w_{(1,0)},w_{(2,2)},w_{(3,6)},w_{(3,7)}\\ 4: w_{(1,0)},w_{(3,4)},w_{(3,5)},w_{(2,3)}\\ 5: w_{(1,0)},w_{(3,4)},w_{(3,5)},w_{(3,6)},w_{(3,7)}\\ 6: w_{(2,0)},w_{(2,1)},w_{(1,1)}\\ 7: w_{(2,0)},w_{(2,1)},w_{(2,2)},w_{(2,3)}\\ 8: w_{(2,0)},w_{(2,1)},w_{(2,2)},w_{(3,6)},w_{(3,7)}\\ 9: w_{(2,0)},w_{(2,1)},w_{(3,4)},w_{(3,5)},w_{(2,3)}\\ 10: w_{(2,0)},w_{(2,1)},w_{(3,4)},w_{(3,5)},w_{(3,6)},w_{(3,7)}\\ 11: w_{(3,0)},w_{(3,1)},w_{(2,1)},w_{(1,1)}\\ 12: w_{(3,0)},w_{(3,1)},w_{(2,1)},w_{(2,2)},w_{(2,3)}\\ 13: w_{(3,0)},w_{(3,1)},w_{(2,1)},w_{(3,4)},w_{(3,5)},w_{(2,3)}\\ 14: w_{(3,0)},w_{(3,1)},w_{(2,1)},w_{(2,2)},w_{(3,6)},w_{(3,7)}\\ 15: w_{(3,0)},w_{(3,1)},w_{(2,1)},w_{(3,4)},w_{(3,5)},w_{(3,6)},w_{(3,7)}\\ 16: w_{(3,0)},w_{(3,1)},w_{(3,2)},w_{(3,3)},w_{(1,1)}\\ 17: w_{(3,0)},w_{(3,1)},w_{(3,2)},w_{(3,3)},w_{(2,2)},w_{(2,3)}\\ 18: w_{(3,0)},w_{(3,1)},w_{(3,2)},w_{(3,3)},w_{(2,2)},w_{(3,6)},w_{(3,7)}\\ 19: w_{(3,0)},w_{(3,1)},w_{(3,2)},w_{(3,3)},w_{(3,4)},w_{(3,5)},w_{(2,3)}\\ 20: w_{(3,0)},w_{(3,1)},w_{(3,2)},w_{(3,3)},w_{(3,4)},w_{(3,5)},w_{(3,6)},w_{(3,7)}\end{cases} \tag{14}$$

3) State

The state for DQN-FRAME, with size $3N_c$+1, is given by,

$$\left[n_{frame},\tilde{N}_{d,0},\tilde{N}_{coll,0},\dots\tilde{N}_{d,c},\tilde{N}_{coll,c},\tilde{\delta}_{\max,c}\dots\right] \tag{15}$$

IV. Results

The algorithm is evaluated in a scenario with a single BS and 18000 terminals taking into account [12]. Part of these terminals (13000) have been assigned a high access priority, while the remaining (5000) have low access priority. The deployment and activation of the terminals is described in section II.C. The priority of terminals is considered by means of μ_0=0.8, μ_1=0.2 in the reward function presented in (8). We assume ideal beams.

Regarding the system parameters: i) there are 54 preambles, ii) the periodicity of RAOs is 10ms, iii) T_{SIB2} is 400 ms, and iv) we define T_{FRAME} = 2 seconds (25×T_{SIB2} or 1000 RAOs). According to [11], there is a maximum number of terminals that can be detected because of the limited resources to communicate the successful detection (related to msg-3 and msg-4). To take into account such limitation, we define that up to 20 terminals can be detected on a single RAO. Beyond that value, the rest are declared as collided. Furthermore, the access request follows the description provided in section II.A

Both DQNs are implemented with two hidden layers and ReLU functions. Their configurations are (9,25,25,64) and (46,64,64,21) for DQN-ACB and DQN-FRAME, respectively. The hierarchical codebook has L = 4 layers, providing the codeword with the highest resolution to be π/4 radians. The hyperparameters are the same in both DQNs: α_{ACB}=0.01, α_{FRAME}=0.0001 (learning rate with Adam solver [13]), γ_{ACB}=0.999, γ_{FRAME}=0.995 (discount factor), τ=0.01 (soft update DQNs). The batch size is 64. The target networks of DQN-ACB and DQN-FRAME are updated every 16 and 8 realizations, respectively.

The proposed algorithm is evaluated over multiple system realizations or episodes. Each one considers the deployment of terminals with given time activation (following the Beta function or uniform depending on the type of terminal). The episode finishes when all terminals are either detected or declared as failure. In the next sub-section we review the convergence of the value function of the RL-based algorithms, how the value of P_{ACB} and frame configuration evolves for a single episode, once the models are trained. Finally, the obtained delays and the number of served terminals will be analyzed. The proposed algorithm can be adjusted for the following configurations:

- **No DRL**: All RAOs apply an omnidirectional antenna pattern (codeword w(0,0)) and ACB factors are set to 1. This strategy is used as a baseline for comparative purposes.
- **DRL-ACB** *(time)*: All the RAOs are using always the codeword w(0,0) and there is not decision on the type of frame. Only DQN-ACB is active. The ACB factors are reset to 1 every T_{FRAME}.
- **DRL-Frame** (*space*): All ACB factors are set to 1, only DQN-Frame is active.
- **DRL-(ACB,Frame)** (*time-space*): Both DQNs are active and trained simultaneously. Notice that every time the frame changes, the decision on the ACB factors in the first T_{SIB2} cannot rely on previous states. In that particular case, the ACB factors are reset to one.

A. Training convergence

The evolution of the obtained cumulative ACB reward during the training process in equation (8) for the algorithms that work in the time domain is presented in Fig.6. We have considered up to 7000 episodes for training where the probability of selecting a random action decreases from ε=1 to ε=0.02 as a function of the episode number. We must emphasize that the configurations No DRL and DRL-FRAME do not optimize the ACB reward. We can observe that DRL-ACB and (DRL-ACB,Frane) tends to converge with the number of episodes.

Fig. 7 presents the cumulative reward of DQN-Frame network (13). Here, the algorithms that maximize the reward are the DRL-Frame and DRL-(ACB, Frame). Both algorithms tend to converge to similar values. However, to compare in a fair way the different protocols we should pay attention to the number of served terminals, which will be tackled in section IV.C.

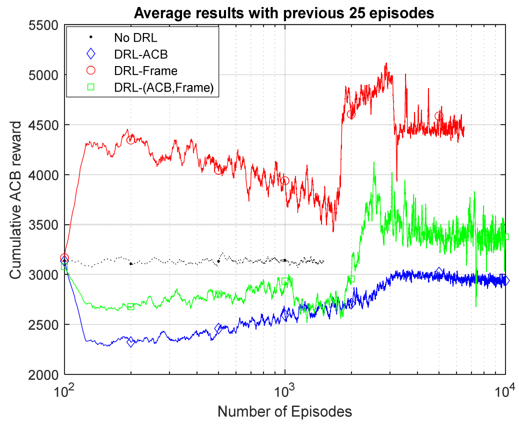

Fig. 6. Cumulative ACB reward averaged over 25 episodes as a function of the episodes considered for training.

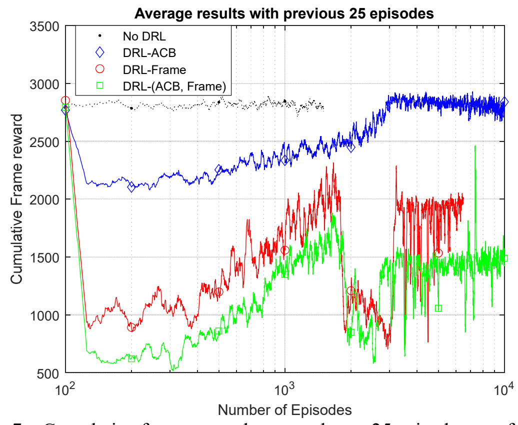

Fig. 7. Cumulative frame reward averaged over 25 episodes as a function of the episodes considered for training

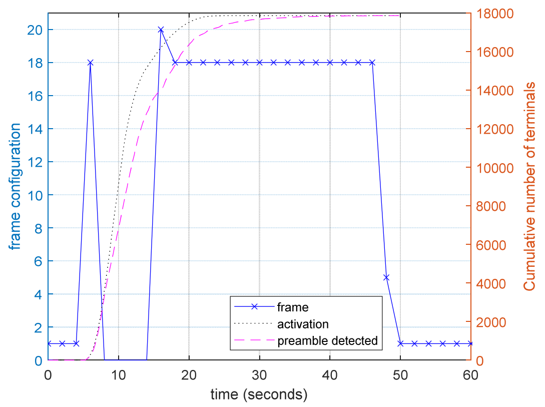

Fig. 8. Evolution of the selected frame, $n_{frame,}$ under the (DRL-ACB,Frame) algorithm for a single episode (left y-axis). Also shown the cumulative number of terminals activated for the first time and cumulative number of detected terminals (right y-axis)

B. Dynamics of the algorithm for a single episode

Once the models for the DQN networks are trained, we evaluate a single episode to review how the frame structure evolves in time depending on the activity of the terminals. Fig. 8 sketches the configuration selected over the different time periods for the algorithm DRL-(ACB, Frame), (14). The dotted lines show the cumulative number of terminals activated in the system, while the dashed lines denotes the cumulative number of successfully detected terminals. In this episode, terminals start to be activated at time t=5 seconds and the last terminal being detected is at time t=50 seconds. When there are many terminals the algorithm tends to select the n_{frame}=18 which assigns the codewords with the largest resolution in the space, so that terminal are divided into subareas with independent barring factors. Having a lower number of terminals per subarea means that a higher barring factor can be used, increasing the number of terminals that succeed in the ACB check. Afterwards, the selected n_{frame}=0 works with low ACB factors, so that the number of terminals access to the network later, that is the reason to select n_{frame}=18 in t=18 s. Once the RA requests decrease, then selected frame structure tends n_{frame}= 1.

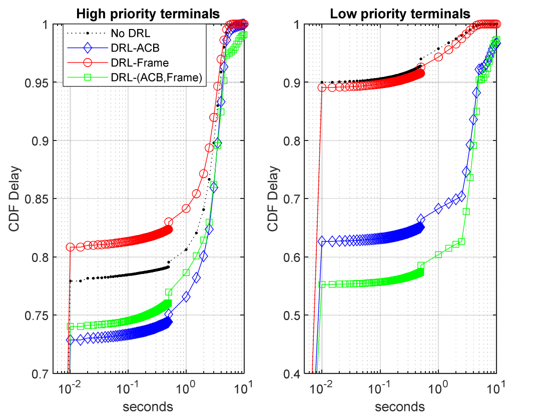

Fig. 9. CDF of the attained delay per terminal under the diferent approaches in 50 episodes with the trained DQN networks. (Left) high priority terminals, (right) low priority terminals.

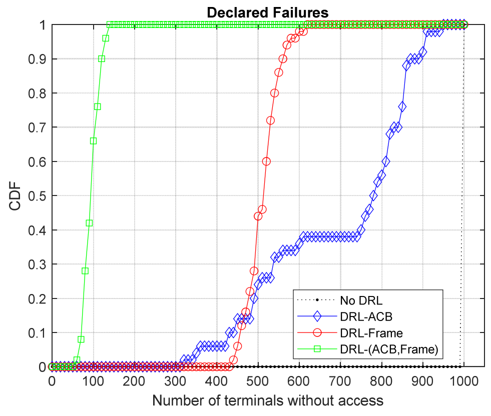

Fig. 10. CDF of the number of terminals that could not be detected because: (left) maximum number of transmissions and (right) the ACB mechanism backlogged the random access form more than 10 times

C. Delay and served terminals

Once the DQN models are properly trained, we consider 50 episodes and measure the attained RA delays, Fig. 9, and the number of terminals that did not succeed, Fig 10. In terms of delay, Fig.9 shows the cumulative density function (CDF) of the experienced delay defined from the first time that a terminal needs to do a RA request until it is successfully detected by the BS. Fig.9-left shows the CDF for the high priority terminals, while Fig.9-right is devoted for low priority ones. We can observe that DRL-(ACB,Frame) algorithm is not the best one, getting similar results than other algorithms. However, those algorithms have a very different performance in terms of served terminals. Fig 10 presents the CDF of terminals that have collided 10 times and cannot access to the network. The DRL-(ACB,Frame) algorithm is the one that is able to guarantee service to more terminals. The remaining algorithms attain better results in terms of delay at the cost of serving fewer terminals.

V. Conclusions

The present work has investigated the benefits of using the spatial domain to deal with the collisions of the random access channel when BS is equipped with multiple antennas. The BS applies a receiver codebook-based beamforming that adapts the beamwidth as a function of the number of detected collisions, by selecting beams with smaller beamwidths. With the objective of adjusting the different parameters and combine with time domain based procedures, i.e. ACB mechanism, we consider the use of reinforcement learning theory with Deep Q Networks. Results have elucidated that the use of the space domain provides a significant benefits in terms of guaranteeing the maximum transmission delay for a large number of terminals.

References

[1] Ericsson, "Ericsson Mobility Report, Nov. 2020, Available on: https://www.ericsson.com/en/mobility-report

[2] I. Leyva-Mayorga et al., "Performance Analysis of Access Class Barring for handling massive M2M traffic in LTE-A Networks", in Proc. IEEE Intl. Conf. on Communications (ICC), 2016

[3] N. Jian et al., "Deep Reinforcement Learning for discrete and continuous massive access control optimization", in Proc. IEEE Intl. Conf. on Communications (ICC), 2020

[4] R.S. Sutton, A.G. Barto, "Reinforcement Learning: An Introduction", Second Edition. The MIT Press, Cambridge, Massachusets, 2020

[5] Z. Chen, D. B. Smith, "Heterogeneous Machine-Type Communications in Cellular Networks: Random Access Optimization by Deep Reinforcement Learning", in Proc. IEEE Intl. Conf. on Communications, 2018

[6] L.Tello-Oquendo et al., "Reinforcement Learning-Based ACB in LTE-A Networks for Handling Massive M2M and H2H Communications", in Proc. IEEE Intl. Conf. on Communications (ICC), 2018

[7] H. Yang, et al., "Deep Reinforcement Learning Based Massive Access Management for Ultra-Reliable Low-Latency Communications", IEEE Trans. on Wireless Comm., vol. 20, no.5, May 2021

[8] Z. Xiao, T. He, P. Xia, X.-G. Xia et al., "Hierarchical codebook design for beamforming training in millimeter-wave communication", IEEE Trans. Wireless Commun., vol. 15, no. 5, pp. 3380-3392, 2016

[9] Y.R. Li et al., "Beam Management in Millimeter-Wave Communications for 5G and Beyond", IEEE Access, Special Section on mmW Comm: New Research trends and challenges, Jan 2020.

[10] V. Mnih et al., "Playing Atari with Deep Reinforcement Learning". Available at https://arxiv.org/pdf/1312.5602.pdf

[11] E.Dahlman, S. Parkwall, J. Skold, "5G NR: The Next Generation Wireless Access Technology", Academic Press, 2020

[12] 3GPP, "Study on RAN improvements for machine-type communications," 3GPP TR 37.868 V11.0.0, Sep. 2011

[13] D.Kingma, J.L. Ba, "ADAM: A Method for Stochastic Optimization", in Proc. Intl. Conf. on Learning Representations (ICLR), 2015