Deep Q-Learning Framework for Improving Spectral Efficiency in D2D Communication

Lucas Baião Pires and Paulo Henrique Portela de Carvalho

***Abstract*—Device-to-device (D2D) communication is expected to play a big role in 5G, in order to enable new applications in the mobile system such as vehicular-communications (V2X) and internet of things (IoT). This should increase the bandwidth demand, which makes higher spectral efficiency ever more desirable. This paper proposes a Deep Q-Learning power allocation framework for maximizing spectral efficiency in D2D communication, in underlay mode, while satisfying the mobile user's quality of service (QoS) requirements.**

***Keywords*—Device-to-device (D2D), 5-th generation (5G), deep reinforcement learning, artificial intelligence.**

I. Introduction

Due to the expected growth in the mobile devices and bandwidth-hungry applications [1], the fifth generation of mobile communications (5G) will need different methods for improving the provided data rate. Device-to-device (D2D) communications is expected to play a key role in internet of things (IoT) applications, such as health care, vehicular communications (V2X), industry automation, and many others. Beyond IoT, D2D may also help with content caching [2], traffic offloading [3] and ad-hoc applications, such as social applications and emergency services [4, 5].

D2D communication makes it possible for two nearby devices to communicate directly with each other in high quality, usually, due to the short distances between the users [6]. In mobile communications, when used in underlay mode, D2D communication may improve the system's spectrum efficiency by reusing the physical resource blocks (RBs) [7]. Additionally, this technique can also bring improvements to energy efficiency and fairness [8, 9, 10].

Despite its benefits, underlay D2D communications brings an intrinsic problem, which is the interference between the mobile user equipment (MUE) and the D2D devices while sharing the same RB. Considering the uplink transmission, there must be a power control method in order to avoid interference from the D2D devices on the base station (BS). The whole idea about D2D communications revolves around enabling the link between D2D pairs without disrupting the link between MUE and the BS.

Power control in mobile communications is a topic that has been extensively studied in the literature. In the last years, its application to D2D communications can be seen in related researches [8, 9, 11, 12]. The community has been approaching this problem with numerical and statistical methods, such as game theory and machine learning.

Lucas Baião Pires (180139541@aluno.unb.br), Paulo H. P. de Carvalho (paulo@ene.unb.br), Departamento de Engenharia Elétrica, Universidade de Brasília - UnB, Brasília - DF;

Recently, Reinforcement Learning (RL), a family of machine learning algorithms, has been applied to many diverse problems, such as robotics [13], video-games [14] and telecommunications. In telecommunications, there are already researches studying RL in diverse fields. Some of these fields are security [15], resource allocation [16, 17], content access request [18] and power allocation [8, 9, 19].

In this paper, we propose a non-cooperative Deep Q-Learning (DQL) power control algorithm for underlay D2D cellular communication, inspired by [8, 9]. The objective is to maximize the system spectral efficiency while guaranteeing the MUE quality of service (QoS) to be over a desired level.

The main contributions of this paper are:

- We propose a new DQL-based power allocation algorithm for the underlay-D2D scenario. DQL has already been proposed for D2D scenario by [9], but only in overlay mode, where there is no concern about the MUE QoS.
- We compare the proposed DQL framework to another D2D power allocation solution, Distributed Q-Learning, which uses traditional tabular Q-Learning (QL), found in [8], and show that the DQL framework is able to achieve better performance and better generalization than Distributed QL. We compare the D2D spectral efficiencies and the MUE availability achieved by both solutions. The MUE availability is a performance indicator proposed here, that is not seen in [8].

The rest of this paper is organized as follows. Section II describes the model and formulates the problem. Section III introduces the theories behind QL and DQL, along with the proposed DQL framework. The simulation results and analysis are presented in Section IV. Finally, section V provides the conclusion of this paper.

II. System Model And Problem Formulation

In our model, we consider a scenario with a single cell, in which D2D users coexist with MUEs. The set of MUEs is denoted by $\boldsymbol{M} = \{1, \cdots, M\}$ and the set of D2D pairs is denoted by $\boldsymbol{N} = \{1, \cdots, N\}$. The D2D pairs and the MUEs are distributed uniformly inside the BS coverage area. We assume the coverage area is circular.

In this work, we consider the uplink transmission, where MUEs and D2D users share the same amount of available RBs. The set of available RBs is given by $\boldsymbol{K} = \{1, \cdots, K\}$. In this situation, there are two types of interference we must worry about. Interference 1 is the interference on the BS, which is caused by the D2D transmitters sharing the RB with the MUE, impacting on the MUE QoS. Interference 2 is the interference

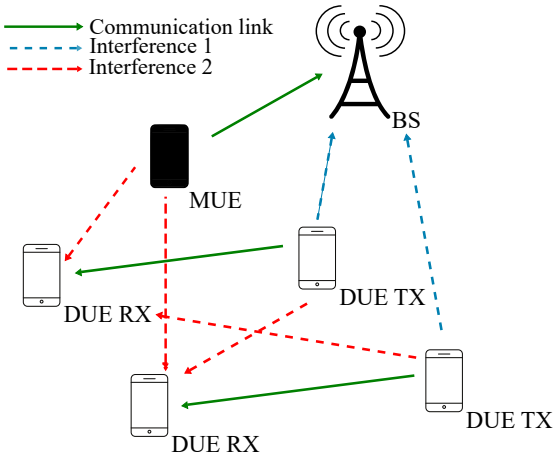

Fig. 1: D2D scenario illustration.

on the D2D receivers. It results from the MUE and other D2D transmitters sharing the same RB and impacts the D2D QoS. This scenario is illustrated by Figure 1.

Inspired by [8], we assume different RBs are orthogonal, allowing us to treat each RB independently.

We measure the system spectral efficiency and QoS using the SINR. The SINR obtained by the i-th D2D user, on the k-th RB, is given by

$$\gamma_k^{d_i} = \frac{p_k^{d_i} \cdot g_k^{d_{ii}}}{\sigma^2 + p_k^m \cdot g_k^{mi} + \sum\limits_{\substack{j \in \boldsymbol{R}_k \\ j \neq i}} p_k^{d_j} \cdot g_k^{d_{ji}}}, i = 1, 2, \cdots, N \quad (1)$$

where $p_k^{d_i}$ and p_k^m denote the i-th D2D pair's transmitter transmission power and the MUE uplink transmission power, respectively, both sharing the k-th RB. Both transmission powers, per RB, are superiorly bounded by p_{max}, which means $p_k^{d_i}, p_k^m \leq p_{max}, \forall i, m \in \boldsymbol{N} \bigcup \boldsymbol{M}$. $\boldsymbol{R}_k$ is the set of D2D pairs sharing the $k-$th RB. The channel gain between the i-th D2D pair devices, on the k-th RB, is given by $g_k^{d_{ii}}$. The channel gain between the MUE and the i-th D2D pair receiver, on the k-th RB, is given by g_k^{mi}. At last, the channel gain between the j-th D2D pair transmitter and the i-th D2D pair receiver, on the k-th RB, is denoted by $g_k^{d_{ji}}$. The noise power is depicted as σ^2. We also have to know the MUE SINR, on the k-th RB, which is

$$\gamma_k^m = \frac{p_k^m \cdot g_k^{m0}}{\sigma^2 + \sum\limits_{j \in \boldsymbol{R}_k} p_k^{d_j} \cdot g_k^{j0}} \quad (2)$$

Following the established convention, the channel gains between the MUE and the BS, and between the j-th D2D transmitter and the BS, both on the k-th RB, are given by g_k^{m0} and g_k^{j0}, respectively.

In our problem, we desire to maximize the D2D pairs' spectral efficiencies [20], while maintaining the MUE QoS at a minimum desired level. For the sake of simplicity, we consider the MUE transmission power and the RB allocation are given and fixed. Therefore, we can write the optimization problem as:

$$\begin{aligned} &\max_{\boldsymbol{p}} \sum_{k=1}^{K} \left\{ \log_2\left(1 + \gamma_k^m\right) + \sum_{i \in \boldsymbol{R}_k} \log_2\left(1 + \gamma_k^{d_i}\right) \right\} \\ &\gamma_k^m \geq \tau_0 \\ &0 \leq p_k^{d_i} \leq p_{max}, \forall i, k \end{aligned} \quad (3)$$

where $\boldsymbol{p} = (p_k^{d_1}, \cdots, p_k^{d_i}, \cdots, p_k^{d_N}), \forall k, i \in \boldsymbol{K}, \boldsymbol{N}$. The optimization problem in (3) consists of finding the set of transmission powers, for each D2D pair on the k-th RB, which will maximize the capacity for the MUE and the D2D pairs, while satisfying two restrictions. The restrictions are the minimum MUE SINR, τ_0, and the maximum transmission power level, p_{max}. In order to solve this problem, we resort to a DQL framework.

III. DQL Framework

In this section, we present the theory behind single agent RL and multi-agent RL. We begin with classical Q-Learning and progress to DQL. In the end, we propose a DQL framework to solve a modified version of the optimization problem in (3), so we may approach the solution in a multi-agent, non-cooperative manner.

A. Single Agent Q-Learning

QL [21] is a popular model-free, off-policy RL algorithm. It seeks to approximate the optimal action-value function, $Q^*(s, a)$, for state $s \in S$ and action $a \in A$. QL works with the tuple $(S, A, T, R(s, a))$, where S is the finite set of the environment states, A is the finite set of agent actions, $T : S \times A \times S \mapsto [0, 1]$ is the state transition probability function and $R : S \times A \times S \mapsto \boldsymbol{R}$ is the reward function. The environment state transitions are considered markovian processes, making the optimization problem a markovian decision process (MDP).

The algorithm makes use of the interaction between an agent i and environment. At time t, the state is given by s_t^i. Then, the agent takes an action, a_t^i, which leads the environment to the next state, s_{t+1}^i, according to the state transition function. This action also returns a reward, r_t^i. The agent chooses its actions following a policy $\pi : S \mapsto A$. Knowing this, we can calculate the state-value function [21],

$$V^\pi(s) = E_\pi\left[r_t | s_t = s\right] = E_\pi\left[\sum_{t=0}^{\infty} \eta^t r_t | s_t = s\right] \quad (4)$$

where $0 \leq \eta \leq 1$ is a discount factor. With (4), we can calculate the action value function,

$$Q(s_t, a_t) = E[r(s_t, a_t) + \lambda V^\pi(s_{t+1})] \quad (5)$$

where $r(s_t, a_t)$ is the received reward at state s, given action a was taken, at instant t, and $0 \leq \lambda \leq 1$ is another discount factor.

Following the Bellman equation [21], we may write the action value function as

$$Q^{\pi}(s,a) = r(s,a) + \lambda \sum_{s' \in S} P(s,a,s') Q^{\pi}(s',a') \tag{6}$$

where $s = s_t, a = a_t, s' = s_{t+1}, a' = a_{t+1}$, and $P(s,a,s')$ is the transition probability from state s to state s' when the action a is taken. Finding the optimum policy is equivalent to finding the optimal action-value function $Q^*(s,a)$ and its maximum values. Therefore, the optimal policy may be described as

$$\pi^*(s) = \arg\max_{a \in A} \{Q^*(s,a)\} \tag{7}$$

which means the optimal policy, for a given state s will return the action a that maximizes $Q^*(s,a)$. Hence, the iterative QL algorithm approximates $Q^*(s,a)$ by

$$\begin{aligned} &Q(s,a) \leftarrow Q(s,a) + \\ &\alpha \left[r(s,a) + \lambda \max_{a'} Q(s',a') - Q(s,a) \right] \end{aligned} \tag{8}$$

where $0 \leq \alpha \leq 1$ is called the learning rate.

B. Deep Q-Learning

Deep Q-Learning (DQL) is a method that implements QL using Deep Learning (DL) [14, 22]. Since neural networks (NN) are proved to be universal function approximators [23], DQL uses DL to approximate the optimal action-value function $Q^*(s,a)$. DQL is able to handle continuous-states and it achieves greater sample efficiency, resulting in a lower variance than traditional QL [24].

RL is known to diverge when implemented with NNs. To address this problem, [14] proposes the experience replay. The experience replay randomizes over the data, removing correlation between observation sequences. Additionally, the action values Q are updated every B iterations, not on every iteration, reducing correlation with the target.

The action-value function is parameterized as $Q(s,a;\theta_l)$, where θ_l represents the NN weights at the l-th iteration. We store the agent's experience $e_t = (s_t, a_t, r_t, s_{t+1})$, at each time step t in a dataset $D_t = \{e_1, \cdots, e_t\}$. The NN uses mini-batches uniformly sampled from D_t for its learning process. The used loss function, at the i-th iteration, is [14]

$$L_i(\theta_l) = E_{(s,a,r,s')} \left[\left(r + \gamma \max_{a'} Q(s',a';\theta_l^-) - Q(s,a;\theta_l)^2 \right) \right] \tag{9}$$

where θ_l and θ_l^- are the Q-Network and the target weights, respectively. The target parameters θ_l^- are only updated with θ_l every B steps, and are kept fixed otherwise.

C. DQL Power Control Framework

The optimization problem in (3) may be decomposed into K-parallel sub-optimal problems, each problem being solved for each RB. The RBs are independent from each other. Considering one RB, the RL algorithm is defined as follows:

Agent: The agents are the D2D transmitters.

State: The state is given by

$$s_k^i = (n_k, d_1^i, d_{2,k}^i, d_{3,k}, I_k),$$

where n_k is the number of D2D pairs accessing the k-th RB, d_1^i is the distance from the i-th agent to the BS, $d_{2,k}^i$ is the distance from the i-th D2D receiver to the MUE on the k-th RB, $d_{3,k}$ is the distance between MUE and BS on the k-th RB, and, at last, I_k is the interference indicator on the k-th RB. The interference indicator is defined as

$$I_k = \begin{cases} 1 & \gamma_k^m \geq \tau_0 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

We assume the agents exchange information with the BS, in order to acquire the input data to the DQL framework. It is interesting to notice this approach would not be practical with tabular QL, since these distances are continuous values. QL would need a discretization process, which would incur in information loss. With DQL, this problem does not exist.

Actions: The agents' actions consists of a set of transmission power levels. It is given by

$$\boldsymbol{A} = (a_1^k, \cdots, a_y^k)$$

where a_y^k represents the power level y for the k-th RB. The chosen training policy was the ϵ-greedy [14, 21]. This policy dictates how the agents will pick their actions during training. It is denoted by

$$\pi(s) = \begin{cases} a_{\text{random}} & \text{with probability } \epsilon \\ \arg\max_a Q(s,a;\theta_i) & \text{with probability } 1-\epsilon \end{cases} \tag{11}$$

Reward function: Inspired by [8], the reward function is given by

$$R_k^i = \begin{cases} \frac{1}{C} \log_2 \left(1 + \gamma_k^{d,i} \right), & \gamma_k^m \geq \tau_0 \\ -1, & \text{otherwise} \end{cases} \tag{12}$$

where C is an arbitrary penalty factor to the reward, and R_k^i is the received reward by agent i on the k-th RB.

This reward function was built to minimize the information exchange between agent and BS and make the solution suitable for distributed training, at the cost of obtaining a sub-optimal solution by maximizing the spectral efficiency of each D2D link, instead of the whole RB spectral efficiency.

In this work, since the agents states are independent and identically distributed [25], we use all agents data to train a single Deep Q-Network (DQN), for the sake of training speed and computer resources. In real life, the DQN may be trained in a distributed way, where each D2D device trains its own DQN, or in a centralized way, where the BS collects data from the agents, trains a single DQN, and deploys the DQN to the D2D devices, afterwards.

The DQL framework algorithm, for one RB, is defined in Algorithm 1.

IV. Simulation And Analysis

In this section, we present the obtained results and compare the DQL Framework to the Distributed QL, proposed by [8]. The used simulation settings are presented in Table I. The parameters are the same as in [8]. We consider a macro-cell with a coverage radius of 500 m. We simulate for only 1 RB,

Algorithm 1: DQL Framework

initialize $\epsilon, \epsilon_{\min}, \delta, B$
for *episode = 1, M* **do**
  for *t=1,T* **do**
    initialize sequence $\boldsymbol{s} := (s_1^1, \cdots, s_1^N)$
    preprocessed sequence
     $\boldsymbol{\phi_1} := (\phi(s_1^1), \cdots, \phi(s_1^N))$
    for $i = 1, N$ **do**
      generate a random number $x \in (0, 1]$
      if $x < \epsilon$ **then**
        select random action a_t^i
      else
        select $a_t^i := \arg\max_a Q(\phi(s_t), a; \theta)$
      end
      execute a_t^i
    end
    if $\epsilon > \epsilon_{\min}$ **then**
      $\epsilon := \epsilon - \delta$
    end
    $\boldsymbol{R_t} := (R_t^1, \cdots, R_t^N)$
    $\boldsymbol{\phi}(s_t) := (\phi_t^1, \cdots, \phi_t^N)$
    $\boldsymbol{a_t} := (a_t^1, \cdots, a_t^N)$
    $\boldsymbol{\phi}(s_{t+1}) := (\phi_{t+1}^1, \cdots, \phi_{t+1}^N)$
    store transition $(\boldsymbol{\phi(s_t)}, \boldsymbol{a_t}, \boldsymbol{a_t}, \boldsymbol{\phi})$ in D
    sample random mini-batches of transitions
     $(\phi_j^n, a_j^n, R_j^n, \phi_{j+1}^n)$ from D
    if *episode terminates at step* $j + 1$ **then**
      $y_j := R_j^n$
    else
      $y_j := R_j^n + \gamma \max_a' Q'(\phi_{j+1}, a'; \theta^-)$
    end
    perform a gradient step on $(y_j - Q(\phi_j, a_j; \theta))^2$
     with respect to θ
    if t mod $B = 0$ **then**
      $Q' = Q$
    end
  end
end

since the same algorithm would only be repeated for each RB, in a multiple RBs situation. We vary the number of D2D pairs accessing the RB from 1 to 10 pairs. There is 1 MUE, which is the RB user with priority. The minimum acceptable MUE SINR is $\tau_0 = 6$ dB. For the reward, we use $C = 80$. For the target NN update, we use $B = 10$. B and C were obtained empirically.

TABLE I: Simulation parameters.

Parameters	Values
$p_{\max}$	23dBm
Noise Power / RB	-116 dBm
D2D pair distance	50m
pathloss model between BS and users	$15.3+37.6\log_{10}(d_{km})$ [dB]
pathloss model between users	$28+40\log_{10}(d_{km})$ [dB]
BS antenna gain	17 dBi
User antenna gain	4 dBi

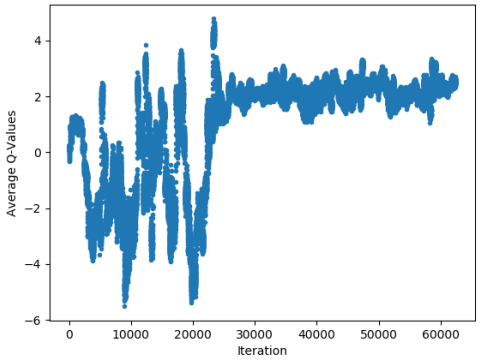

Fig. 2: DQL learning curve.

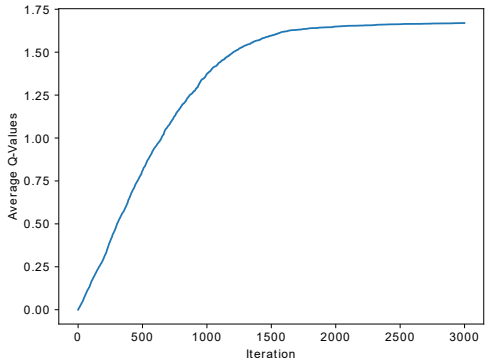

Fig. 3: Distributed QL learning curve.

Figures 2 and 3 depict the DQL and QL learning curves, respectively. They were obtained by taking the averages of the Q-values given by both algorithms, on every learning iteration. We can see DQL takes much longer to converge than traditional QL. Hence, the DQL solution is thought for stationary situations, since dynamic phenomena become stationary over time. In the case of environment changes, the DQL algorithm may be re-trained.

However, DQL offers a great advantage over QL. Once trained, DQL is able to handle many different users' positions distributions and different numbers of users, and even extrapolate for situations it has not seen during training, in contrast with QL, which needs to be re-trained every time the devices change positions, or when the number of devices changes.

In order to measure how well the power allocation respects the MUE QoS requirements, a MUE success indicator is proposed, which measures the rate of how many times the MUE SINR stood above τ_0 across the iterations. It can also be viewed as the MUE link availability.

Figure 4 presents the DQL performance. It shows the total D2D average spectral efficiency and the average MUE success indicator, varying with N. We can see DQL is able to handle varying numbers of D2D pairs, achieving spectral efficiencies that are over 20% higher than the ones achieved by QL, while maintaining the MUE QoS. Looking at Figure 5, the QL solution may achieve high spectral efficiencies. However,

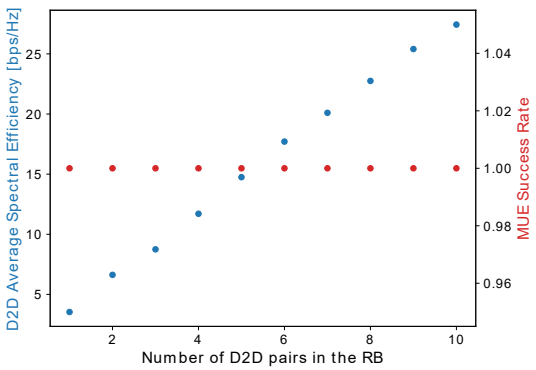

Fig. 4: DQL Framework performance.

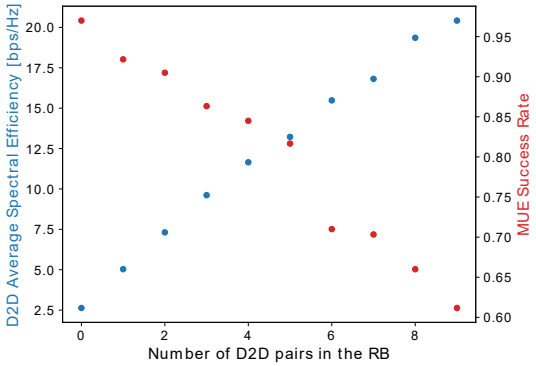

Fig. 5: Distributed QL performance [8].

the MUE availability is much worse, dropping as low as 60%, when $N = 10$, which makes this solution impracticable for implementation in real life.

DQL is able to surpass QL thanks to its greater sample efficiency, lower variance [24], and the possibility of using continuous-states values [23].

V. Conclusion

This paper presented a Deep Reinforcement Learning (DRL) framework to optimize the D2D spectral efficiency and MUE QoS, in D2D communications. We show DQL is able to provide a higher D2D throughput and MUE QoS than standard QL, while generalizing for a wide range of situations. In our results, we also measured the MUE QoS, which is not done by other works. In the future, we plan to increase the simulated scenarios complexity, by adding more devices, RBs, cells and dynamich channels, bringing the simulations closer to real environments.

References

[1] L. Chettri and R. Bera, "A comprehensive survey on internet of things (iot) towards 5g wireless systems," *IEEE Internet of Things Journal*, 2019.

[2] M. Gregori, J. Gómez-Vilardebó, J. Matamoros, and D. Gündüz, "Wireless content caching for small cell and d2d networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1222–1234, 2016.

[3] G. Zhao, S. Chen, L. Qi, L. Zhao, and L. Hanzo, "Mobile-traffic-aware offloading for energy-and spectral-efficient large-scale d2d-enabled cellular networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3251–3264, 2019.

[4] X. Lin, J. G. Andrews, A. Ghosh, and R. Ratasuk, "An overview of 3gpp device-to-device proximity services," *IEEE Communications Magazine*, vol. 52, no. 4, pp. 40–48, 2014.

[5] G. T. . V. 0.0, "3rd generation partnership project; technical specification group sa; feasibility study for proximity services (prose)(release 12)," 2012.

[6] Y. Kai, J. Wang, H. Zhu, and J. Wang, "Resource allocation and performance analysis of cellular-assisted ofdma device-to-device communications," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 416–431, 2018.

[7] K. Zia, N. Javed, M. N. Sial, S. Ahmed, H. Iram, and A. A. Pirzada, "A survey of conventional and artificial intelligence/learning based resource allocation and interference mitigation schemes in d2d enabled networks," *arXiv preprint arXiv:1809.08748*, 2018.

[8] S. Nie, Z. Fan, M. Zhao, X. Gu, and L. Zhang, "Q-learning based power control algorithm for d2d communication," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2016, pp. 1–6.

[9] K. K. Nguyen, T. Q. Duong, N. A. Vien, N.-A. Le-Khac, and M.-N. Nguyen, "Non-cooperative energy efficient power allocation game in d2d communication: A multi-agent deep reinforcement learning approach," *IEEE Access*, vol. 7, pp. 100 480–100 490, 2019.

[10] S. Sharma and B. Singh, "Weighted cooperative reinforcement learning-based energy-efficient autonomous resource selection strategy for underlay d2d communication," *IET Communications*, vol. 13, no. 14, pp. 2078–2087, 2019.

[11] C. Sun, M. Peng, Y. Sun, Y. Li, and J. Jiang, "Distributed power control for device-to-device network using stackelberg game," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2014, pp. 1344–1349.

[12] R. M. Amorim, R. D. Vieira, and P. H. P. Carvalho, "Network efficiency maximization exploiting wcdma spectrum nature for enhancement of d2d underlaying communications," *Analog Integrated Circuits and Signal Processing*, vol. 78, no. 3, pp. 741–752, 2014.

[13] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 253–279, 2019.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[15] X. Liu, Y. Xu, L. Jia, Q. Wu, and A. Anpalagan, "Anti-jamming communications using spectrum waterfall: A deep reinforcement learning approach," *IEEE Communications Letters*, vol. 22, no. 5, pp. 998–1001, 2018.

[16] K. Zia, N. Javed, M. N. Sial, S. Ahmed, and F. Pervez, "Multi-agent rl based user-centric spectrum allocation scheme in d2d enabled hetnets," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, 2018, pp. 1–6.

[17] P. H. P. de Carvalho, R. D. Vieira, and J. P. Leite, "A continuous-state reinforcement learning strategy for link adaptation in ofdm wireless systems," *Journal of Communication and Information Systems*, vol. 30, no. 1, 2015.

[18] N. Kumar, S. N. Swain, and C. S. R. Murthy, "A novel distributed q-learning based resource reservation framework for facilitating d2d content access requests in lte-a networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 718–731, 2018.

[19] J. Xu, X. Gu, and Z. Fan, "D2d power control based on hierarchical extreme learning machine," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2018, pp. 1–7.

[20] J. G. Proakis and M. Salehi, *Digital communications*. McGraw-hill New York, 2001, vol. 4.

[21] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[23] P. Palanisamy, *Hands-On Intelligent Agents with OpenAI Gym: Your guide to developing AI agents using deep reinforcement learning*. Packt Publishing Ltd, 2018.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[25] A. Leon-Garcia, *Probability and random processes for electrical engineering*. Pearson Education India, 1994.