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

Received 13 February 2023; revised 1 June 2023, 25 September 2023, and 23 November 2023; accepted 11 December 2023.
Date of publication 19 December 2023; date of current version 29 December 2023.

The associate editor coordinating the review of this article and approving it for publication was E. E. Tsiropoulou.

Digital Object Identifier 10.1109/TMLCN.2023.3345273

Learning Random Access Schemes for Massive Machine-Type Communication With MARL

MUHAMMAD AWAIS JADOON[1] (Student Member, IEEE),
ADRIANO PASTORE[1] (Senior Member, IEEE),
MONICA NAVARRO[1] (Senior Member, IEEE),
AND ALVARO VALCARCE[2] (Senior Member, IEEE)

[1]Centre Tecnològic de Telecomunicacions de Catalunya (CTTC)/CERCA, 08860 Castelldefels, Barcelona
[2]Department of Radio Systems Research and AI, Nokia Bell-Labs France, 91620 Massy, France

CORRESPONDING AUTHOR: M. A. JADOON (mjadoon@cttc.es)

This work was supported by the European Union H2020 Research and Innovation Programme through Marie Skłodowska Curie action (MSCA-ITN-ETN 813999 WINDMILL), and Grant PID2021-128373OB-I00 funded by Ministerio de Ciencia e Innovación (MCIN)/Agencia Estatal de Investigació (AEI)/10.13039/501100011033 and by "The European Regional Development Fund (ERDF) A way of making Europe."

ABSTRACT This paper investigates various multi-agent reinforcement learning (MARL) techniques for designing grant-free random access (RA) schemes suitable for low-complexity, low-power battery-operated devices in massive machine-type communication (mMTC). Previous studies on RA with MARL have shown limitations in terms of scalability and suitability for mMTC. To address scalability and practicality of the proposed methods, we examine the impact of excluding agent identification in the observation vector of each agent on network performance. We employ value decomposition networks (VDN) and QMIX algorithms with parameter sharing (PS) and compare their policies with the deep recurrent Q-network (DRQN). Our simulation results demonstrate that the MARL-based RA schemes can achieve a better throughput-fairness trade-off between agents without having to condition on the agent identifiers. We also present a correlated traffic model, which is more descriptive of mMTC scenarios, and show that the proposed algorithm can easily adapt to traffic non-stationarities. Moreover, the robustness of the proposed method in terms of scalability is also shown through simulations.

INDEX TERMS Massive machine-type communications, MARL, reinforcement learning, grant-free random access, scalability.

I. INTRODUCTION

THE mMTC paradigm is a key component of 5G and will continue to be important in the development of 6G technologies [1]. As the number of Internet of Things (IoT) devices grows, millions of devices with characteristics different from human-type communication, will require connectivity [2], [3]. To support mMTC in LTE-A, 3GPP has developed narrowband IoT (NB-IoT) and LTE-Machine-type communication (LTE-M) [4], which fall under the category of low power wide area networks (LPWANs). In addition to these cellular standards, non-cellular LPWAN standards such as Sigfox [5] and LoRa [6] have also been commercialized. 3GPP's recent Rel-17 introduces 'NR-Light', a new class of devices that is more capable than NB-IoT or LTE-M but supports different features with a bandwidth larger than NB-IoT/LTE-M but smaller than 5G NR devices. In this paper, we focus on low-power, low-complexity machine-type devices (MTDs) with low data rates (around 1-100 Kbps) and where the communication is mostly uplink dominated. These devices are low-cost, battery-operated and their activity is sporadic. Managing medium access for these devices is challenging, and future wireless technologies will be required to provide massive connectivity to such users.

For devices having above-mentioned characteristics, grant-free RA schemes are preferred, as scheduled access incurs huge signaling overhead [7]. However, RA schemes are prone to collisions and scale poorly. Traditionally, RA schemes such as exponential backoff (EB) [8] employ back-off mechanism at each device to update their transmit probabilities based on feedback from the receiver. These schemes are relatively

simple and decentralized in nature, but their performance is dependent on various assumptions such as the traffic arrival process and whether devices' buffers are saturated. Additionally, the optimal back-off factor for different system parameters is not fixed and may vary [9]. One drawback of EB schemes such as binary exponential backoff (BEB) is the *capture effect*, where a group of devices occupy the channel for a period of time, causing other devices to be deprived of access, hence making the technique unfair. Our goal is to design RA schemes for mMTC that not only provide better throughput but are also fair.

Reinforcement learning (RL) algorithms have become a popular tool for learning RA policies in wireless networks. These algorithms can adapt to changes in the environment and use history to learn the transmission probabilities of devices in a decentralized manner. The MARL has several advantages over traditional EB backoff policies, as it allows for the design of multi-objective policies in a decentralized manner, which is not analytically tractable using traditional methods. However, many MARL solutions such as [10], [11], [12], and [13] are not tailored to the traffic and device characteristics of mMTC systems (details in Section V), and they also struggle with scalability which is a major concern in RA schemes for mMTC. This is because mMTC devices often have low computational power and rely on battery power, making it impractical to perform learning at each device. To the best of our knowledge, the scalability issue has not been adequately addressed in previous RL or MARL studies on RA schemes, and it is unclear how these MARL algorithms can be scaled and whether they are practical and suitable for designing RA policies.

Therefore, one objective of this paper is to design grant-free RA schemes for mMTC using MARL to achieve fairness, adaptability to changes in traffic, and scalability to a large number of devices. Another objective is to provide an analysis by considering the limitations imposed by mMTC users, may be used as a guideline for designing RA protocols. Our contributions are listed in the following.

- A system model is proposed to learn schemes in which the devices can leave and join the network randomly. We do not assume that the devices in the network always have a packet to transmit as opposed to most of the other works for RA with RL, e.g., [10] and [12].
- The proposed system model uses broadcast feedback to reduce the signaling overhead. We assume that the feedback is only sent to the active devices at a given time and the inactive devices are not required to listen to the feedback signal.
- We present a brief report on suitability of MARL algorithms for an mMTC system. Since we want a policy for the devices that can be learned in a centralized training and decentralized execution (CTDE) manner; we provide a comparison between some well-known MARL algorithms and how they may or may not be suitable for our environment. We propose VDN and QMIX algorithms to achieve our objectives. We present our simulation results for VDN and compare it with QMIX and DRQN policies.
- Most of the MARL algorithms that employ CTDE, include an agent-specific identifier into the observation vector of the agents. In case of mMTC, the devices should be able to leave/join the network and the policies should be scalable to a large number of devices. For these reasons, incorporating agent/device identification (ID) is not feasible. We will also show that how the algorithm distributes resources among MTDs fairly when agent IDs are not incorporated and how the algorithms learn an unfair policy when we use agent IDs. However, this way of training may not be suitable in terms of convergence.
- We present our results for *regular* or periodic traffic arrival, in which each MTD receives packets following a random process independently. In addition, we present a correlated traffic arrival model in Section IV, that is more suitable for mMTC system. In the correlated traffic arrival model, the devices follow both regular traffic arrivals and event-driven (ED) traffic arrival. In ED, that is independent of the regular traffic arrival, a subset of MTDs become active together whenever an event happens. We show that our proposed algorithm adapts to different traffic conditions.

II. RELATED WORK

The application of RL to channel access problems in wireless communications goes back to 2010 that used tabular Q-learning [14]. However, it has become popular in the recent years due to the advancements in deep reinforcement learning (DRL). In [15], the authors considered the problem of multiple access where the agents are the base stations to predict the future state of the system. They use recurrent neural network (RNN) and REINFORCE algorithm to learn policies for each agent. In [16], the ALOHA-Q protocol is proposed for a single channel slotted ALOHA scheme that uses an expert-based approach in RL. The goal in that work is for nodes to learn in which time slots the likelihood of packet collisions is reduced. However, the ALOHA-Q depends on the frame structure and each user keeps and updates a separate policy for each time slot in the frame. In [17], the ALOHA-Q is enhanced by removing the frame structure. However, every user still has to keep the number of policies equal to the time slots window it is going to transmit in. Other works such as [10], [11], [12], and [13] consider RL-based multiple access works for multiple channels. In [10], [12], and [18] deep Q-network (DQN) algorithm is used for multiple user and multiple channel wireless networks. In [11], another DRL algorithm known as *actor-critic* DRL is used for dynamic channel access. All these works train agents with the assumption that every device has always a packet in its buffer (saturation state). Moreover, it is not clear whether their algorithms can be scaled for higher number of agents. Interestingly, these works also do not compare their results with any backoff techniques such as EB to show whether

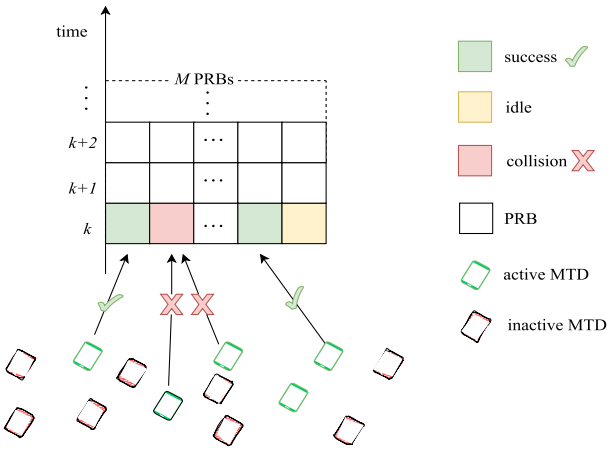


FIGURE 1. System model.

their results outperform them. In [19], authors propose a RA procedure for delay sensitive applications using the context IDs of the devices along with the two-step RA procedure. This is done by predicting the traffic of the devices. A RA strategy for initial access (4 message exchange) to allocate resources in proposed in [20]. They assume that each device also reports its energy levels and access delay to the centralized receiver. Therefore, signaling overhead in this work is high for massive access and it is not energy efficient.

Recently, a RA protocol for initial access is proposed in [21] where results were shown for both regular and bursty traffic arrivals. In [22], RL-based strategy is proposed considering the correlated traffic model. In our previous work [23], we had used DQN with a single resource for the devices following Poisson process for traffic arrival and in [24] we showed how DQN with PS is scalable for bursty traffic arrival model. We minimized the collision rate for an energy efficient policy by penalizing the collisions. Furthermore, the packet delay is also shown to be reduced with the proposed algorithm. To show the effectiveness of DRL in learning new access strategies, in [25], a heterogeneous environment is considered in which an RL agent learns an access scheme in co-existence with slotted ALOHA and a time division multiple access (TDMA) access scheme. In [26], access class barring (ACB) mechanism has been optimized for NB-IoT using DRL. A multiple access algorithm is designed using actor-critic MARL in [27].

III. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a synchronous time-slotted wireless network with a set $\mathcal{N} = \{1, \ldots, N\}$ of MTDs, a set $\mathcal{M} = \{1, \ldots, M\}$ of shared orthogonal physical resource blocks (PRBs) and a receiver as shown in Fig. 1. The physical time is divided into slots, each of duration 1 and the slot index is $k \in \mathbb{N}$. At each time slot, we assume that only $\mathcal{N}_a \subseteq \mathcal{N}$ devices are active, and the activity pattern follows a random process. Each active MTD transmits over the shared PRBs in a grant-free manner. At each time slot k, an MTD can transmit only one packet and it can transmit it only over one resource $m \in \mathcal{M}$. The MTDs are assumed to have perfect synchronization. Moreover, each MTD is equipped with a buffer to store the packets in its queue and each device n can only store at most one packet. The buffer state at time k is defined as $B_n(k) \in \{0, 1\}$, where $B_n(k) = 1$ if there is a packet in the buffer and it is 0 otherwise. If the buffer $B_n(k)$ is full, new packets arriving at device n are discarded and are considered lost. Each device becomes active whenever a packet is generated at the device following one of the traffic arrival models given in Section IV. At each time slot k, MTD n takes an action,

$$A_n(k) \in \mathcal{A} = \{0, 1, \ldots, M\}, \tag{1}$$

where $A_n(k) = 0$ corresponds to the event when user n chooses to not transmit and $A_n(k) = m$ corresponds to the event when user n transmits a single packet on channel m for $1 \leq m \leq M$. If only one user transmits on the channel m in each time slot k, the transmission is successful, whereas a collision event happens if two or more devices transmit in the same time slot. The collided packets are discarded and need to be retransmitted until they are successfully received at the receiver.

For feedback, we consider a broadcast feedback signal $F(k)$ from the receiver that is common to all the devices. Formally, we define

$$F(k) = \{F_1(k), \ldots, F_M(k)\}, \tag{2}$$

and $F_m(k)$ stands for the feedback corresponding to the channel m and for each time slot k, it is defined as

$$F_m(k) = \begin{cases} 1 & \text{if success at time slot } k \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Let us define the binary set $\mathcal{B} = \{0, 1\}$. We define *success* the event $G_{n,m}(k) \in \mathcal{B}$ for user n as a function of the feedback $F_m(k)$ and $A_n(k)$, i.e., $g : (F_m(k), A_n(k)) \mapsto G_{n,m}(k)$, and it is locally computed by each device. We define $G_{n,m}(k)$ for device n and $\forall m \in \mathcal{M}$ as,

$$G_{n,m}(k) = \begin{cases} 1 & \text{if } A_n(k) = m \text{ and } F_m(k) = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Since each device can only transmit on one resource at each time slot, the indicator whether the transmission on any resource for the device n has been successful or not, can be written as

$$G_n(k) = \sum_{m=1}^{M} G_{n,m}(k) \in \{0, 1\}. \tag{5}$$

Furthermore, we define the matrix with N rows and M columns for success events as,

$$\boldsymbol{G} = \left(G_{n,m}\right) \in \mathcal{B}^{N \times M}. \tag{6}$$

We assume that each user keeps a record of its previous actions, feedback and its current buffer state $B_n(k)$ up to h past

instants, where we refer to h as the *history length*. Therefore, the tuple

$$S_n^0(k) = \big(A_n(k-h), \ldots, A_n(k-1), \quad F(k-h), \ldots, F(k-1), B_n(k)\big), \tag{7}$$

is referred to as the *state* of user n at time k for IDs $= 0$ case. For IDs $= 1$ case, we write the state of the agent n as

$$S_n^1(k) = \big(I_n, S_n^0(k)\big), \tag{8}$$

where I_n is a *one-hot encoded* vector of size N for device n.

We will use a general notation $S_n(k)$ to denote the state of a device n for both cases unless mentioned otherwise and we define $S(k) = \big(S_1(k), \ldots, S_N(k)\big)$ as the *global history* of the system.

The feedback signal $F(k)$ is only recorded by the devices that are active at time $k-1$. If a new device becomes active at time k, its state is initialized with $A_n(k-1) = 0$, and $F_m(k-1) = 0, \forall m \in \mathcal{A}$ for its local history $S_n(k)$. The memory is initialized with zero values. Moreover, we set zero values for the time a device has been inactive if the time of inactivity is smaller than the history size.

Definition 1: A policy or access scheme of user n at time slot k, is a mapping from $S_n(k)$ to a conditional probability mass function $\pi_n(\cdot|S_n(k))$ over the action space $\mathcal{A}$. We consider a distributed setting in which there is no coordination or message exchange between users for the channel access. Each new action $A_n(k) \in \{0, 1\}$ is drawn at random from $\pi_n(\cdot|S_n(k))$ as follows:

$$\Pr\big\{A_n(k) = a \;\big|\; S_n(k) = s\big\} = \pi_n(a|s). \tag{9}$$

We are interested in developing a distributed transmission policy for slotted RA that can effectively adapt to changes in the traffic arrivals and provide better performance in terms of throughput, latency, and fairness than the baseline reference schemes. We consider EB policies as our baseline schemes. More specifically, we use BEB when the value of backoff factor is 2, which has been used in IEEE 802.11 and IEEE 802.3 standards.

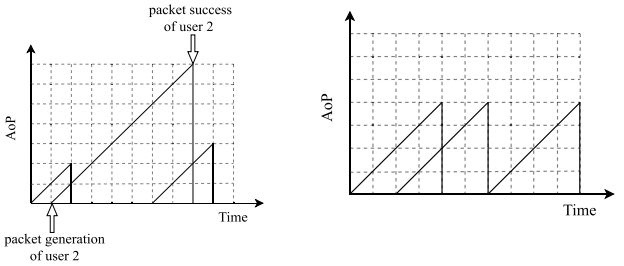
A. PERFORMANCE METRICS

1) THROUGHPUT

The channel throughput is defined as the average number of packets that are successfully transmitted from all the devices divided by the total number of PRBs, over a time window of size K. For the finite time horizon K and for M orthogonal resources, the average throughput of the system is defined as

$$T = \frac{1}{MK} \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{n \in \mathcal{N}} G_{n,m}(k). \tag{10}$$

where $G_{n,m}(k)$ refers to the success event over channel m and $T \in [0, 1]$.



(a) Unfair Scheme (b) Fair Scheme

FIGURE 2. An example of using AoP for fairness for 3 devices, where each device generates a packet.

2) AGE OF PACKETS

The age of packet (AoP)[1] of device n, denoted as $w_n(k)$, grows linearly with time if a packet stays in the buffer of the device, and it is reset to 0 if the packet is transmitted successfully. Specifically, we assume that $w_n(1) = 0$, and the AoP $w_n(k)$ evolves over time as follows:

$$w_n(k) = \begin{cases} 0 & \text{if } B_n(k) = 0 \\ w_n(k-1) + 1 & \text{otherwise.} \end{cases} \tag{11}$$

The average AoP for user n after a time span of K time slots is given by

$$\Delta_n = \frac{1}{K} \sum_{k=1}^{K} w_n(k) \tag{12}$$

and the average AoP of the overall system by $\Delta = \frac{1}{N} \sum_n \Delta_n$. Since techniques such as EB incur *capture effect* [8] where a transmitting device keeps transmitting on the channel for some time, introducing short-term unfairness. In this work, we use the average AoP to measure fairness as well as the average delay budget of the packets. A higher AoP means the scheme is more unfair and has the higher delay and vice versa. To illustrate the concept of fairness with AoP, let us consider an example of 3 users where each user generates a single packet that is successfully transmitted within $K = 10$ time slots as depicted in Fig. 2. The *average* delay (number of time slots taken by each user to transmit their packet is same for both fair and unfair schemes, which is 4 time slots. However, the scheme shown in Fig. 2a is clearly not fair since user 2 takes much more time slots to send its packet as compared to the other two users. This short-term unfairness can be captured with the average AoP and we see that the average for the scheme in Fig. 2a is higher i.e., 1.23) than the one shown in Fig. 2b which is 1.00.

IV. TRAFFIC MODELS

A. REGULAR TRAFFIC MODEL

In traditional RA schemes, the activation of MTDs and the traffic arrival for each user is usually modeled by an independent process. We call such traffic arrival as *regular* traffic

[1]This metric has a different connotation to age of information (AoI).

arrival. Each device follows independent Bernoulli process with average arrival rate λ_n to generate packets in regular traffic model. The average arrival rate for the system for the regular traffic arrival case can then be written as

$$\lambda = \sum_n \lambda_n. \tag{13}$$

However, for mMTC, it is highly likely that some devices are correlated in terms of activation, i.e., some devices observe the same physical phenomenon and activate together. For instance, in industrial fault detection or fleet management, some MTDs are highly likely to transmit at the same time due to the activation of certain event. For instance, in flood or quake detection or land sliding, there is a high chance that devices closer to the event will start transmitting at once. Several recent works have used a correlated activity model to design access schemes for machine-type communication (MTC) [22], [28], [29], [30]. Therefore, the assumption of independent traffic arrival is not valid in this case. Moreover, apart from correlated ED device activity, each MTD also follows regular traffic model [31].

B. CORRELATED TRAFFIC MODEL

The correlated traffic model is a mix of the *regular* traffic and *ED* traffic or *alarm* traffic as depicted in Fig. 3. We assume that the regular traffic generation for each MTD follows an independent random process such as Bernoulli process. Similarly, the ED traffic also follows a random process on top of the regular traffic arrival process. For ED traffic, certain devices are spatially correlated and the ED traffic generation for such devices is dependent on the occurrence of an event in their vicinity. Regular traffic arrival and ED traffic arrival processes are independent of each other.

To formulate this behaviour, we assume that N MTDs are uniformly distributed in a given area. Each MTD can either be in a regular state or alarm state, when active. We consider L event epicentres that are scattered randomly and independently across the given area. The location of the devices is represented by $\mathbf{x} \in \mathbb{R}^2$ and the location of the epicenter of the events is denoted by $\mathbf{y} \in \mathbb{R}^2$. We assume that all MTDs are stationary and fixed to their locations or exhibit very low mobility.

Let $E_{\mathbf{xy}}$ denote the event when a device $n \in \mathcal{N}$ at location $\mathbf{x}$ is triggered into alarm or ED mode by the activation of an event with its epicenter at location $\mathbf{y}$. Let $\bar{E}_{\mathbf{xy}}$ be the complement of $E_{\mathbf{xy}}$ and $p_{\mathbf{xy}}$ denotes the probability of a device n at location $\mathbf{x}$ being triggered into alarm mode by the activation of event at location $\mathbf{y}$. Moreover, we define the probability of device at location $\mathbf{x}$ being in ED mode is $p_{\mathbf{x}}$. We write,

$$\begin{aligned} p_{\mathbf{x}} &= \Pr\big[\text{At least one event triggers MTD at } \mathbf{x}\big] \\ &= 1 - \Pr\big[\text{No event triggers MTD at } \mathbf{x}\big] \\ &= 1 - \prod_{\mathbf{y}\in\mathcal{L}} \Pr\big[\bar{E}_{\mathbf{xy}}\big] \qquad (14) \\ &= 1 - \prod_{\mathbf{y}\in\mathcal{L}} (1 - p_{\mathbf{xy}}), \qquad (15) \end{aligned}$$

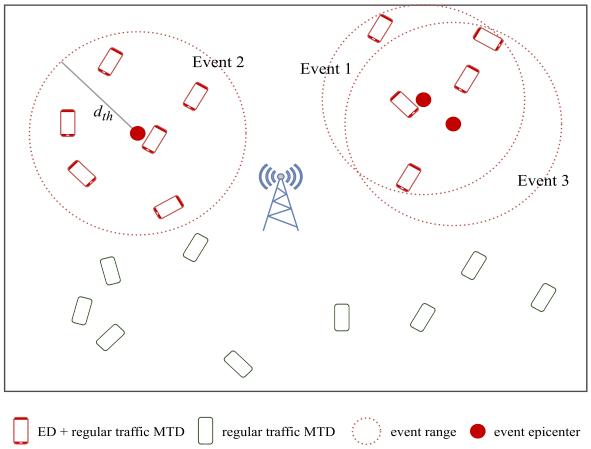


FIGURE 3. MTC Network depicting $N = 20$ MTDs uniformly distributed in a rectangular area with $L = 3$ event epicenters. MTDs follow regular traffic and those within the range of the event epicenter follow both regular and ED traffic.

where we have assumed that events are triggered independently of each other.

Therefore, for the correlated traffic arrival model, each MTD can either be at *alarm* (ED) state or *regular* state for a given time slot k. We denote by $V_{\mathbf{x}}$ the state of device at location $\mathbf{x}$ and we model the states at each time slot k by i.i.d Bernoulli random variable as,

$$V_{\mathbf{x}}(k) = \begin{cases} \text{Regular with prob. } 1 - p_{\mathbf{x}} \\ \text{Alarm with prob. } p_{\mathbf{x}} \end{cases} \tag{16}$$

Moreover, we define with p the probability of an event being active at location $\mathbf{y}$. We assume that each event at epicenter $\mathbf{y}$ triggers a subset $\mathcal{N}_{\mathbf{y}} \subseteq \{1, \ldots, N\}$ of devices. The probability of a device n going into alarm mode depends on the distance of the device from the epicenter $\mathbf{y}$ of the event. Furthermore, each MTD n can sense and report multiple events but at any given time, we assume that it can report about only one event. The MTDs are unaware of the actions, and events sensed by other devices.

V. SUITABILITY OF MARL FOR RA SCHEME DESIGN

To design a RA policy for the multiuser MTC environment, it is important to consider the suitability of MARL algorithms in general for scalability and in particular for the specific characteristics of MTC system. There exists a large body of the literature for medium access using RL in wireless networks but only a few addresses the issues of scalability (e.g., [41] and our recent work [24]). The distributed multiple schemes designed with MARL do have scalability challenges and this is even further exacerbated by the limitations of the mMTC. The MTC system presents the following challenges for MARL algorithms in designing a distributed RA policy:

1) Since the MTDs are low-powered and low-complexity devices that are battery-operated, it is not feasible to perform learning on the devices and therefore, centralized

TABLE 1. Suitability of some popular MARL for designing RA schemes for MTC systems.

MARL Algorithm	Features	Limitations
DQN [32] and variants	Can be used as independent Q learning (IQL) or centralized learning using PS by extending single agent network to multiple agents.	Has convergence issues and it is nearly impossible to learn optimal policy for a large number of agents with PS.
MADDPG [33] MAPPO [34]	• They have a shared critic and local actors. • MAPPO is on-policy and MADDPG is off-policy for CTDE method. • Both circumvent the challenge of non-Markovian and non-stationary environments during learning. • Stabilize learning, due to reduced variance in the value function estimates.	• Need centralized critic which is not scalable to a large number of agents. • The state-space dimensionality grows exponentially for the critic as the number of agents increases. • Most critically, the accumulated noises by the exploratory actions of other agents make the Q-function learning no longer feasible [35].
COMA [36]	• Tackles the credit assignment problem. • Calculates the advantage function which is able to marginalize a single agent's actions while keeping others fixed.	
QMIX [35] VDN [37]	• Lie between COMA and IQL. • Better scalability than centralized critic methods. • Each agent has its own network and then mixing network takes the Q-values of agents to measure Q_{tot}. • QMIX uses NNs to make the function monotonic whilst VDN is the linear combination of the Q values of each agent. • Both make use of RNNs.	• Scalability to a massive number of agents. • Different policy for each agent and it can also be applied to homogeneous agents with PS. • The information might be lost in this way of centralized training. • It is not sure whether the training with PS without using agent IDs will result in better policy than DQN or not.
Mean Field MARL (MF-MARL) [38]	Good scalability results and and it considers the effects of neighbouring agents for each agent to estimate its value function	Requires communication between agents; each agent has its own policy to learn.
Multi-Actor-Attention-Critic (MAAC) [39]	Uses attention mechanism to incorporate the effects of important agents whom actions are given more consideration than others	Requires communication among agents, scales better than MADDPG but their results are only for a small number of agents.
Soft Actor Critic [40]	• Extension of AC methods. Uses the notion of entropy to encourage exploration and to avoid converging to non-optimal policies. • Can be used for multiagent system with PS, local actors and local critics.	• Has convergence issues, like DQN. • The performance is not known with centralized training, i.e., centralized critic and centralized actor, an extension of single agent system just like DQN.

training and decentralized execution (CTDE) method is required for learning.

2) Usually, PS method is used for homogeneous agents, and each agent's ID is used in the observation (state) vector to distinguish between agents. The homogeneous agents are those that have the same state-space and action-space. In our system model, since the devices[2] have variable sleep cycles and they should be able to join/leave the network, it is not feasible to use agent identification. Therefore, we require each agent to have a single policy using PS but at the same time, without using agent IDs.
3) Any communication to exchange channel or device information between MTDs is not energy efficient as it will drain the battery of the devices. Therefore, the devices do not communicate with each other, and they do not know the actions taken by the other devices.

[2]The terms agent, device and MTD are used interchangeably throughout the paper.

In Table 1, we provide the comparison of some popular MARL algorithms and their suitability to the proposed system model. Even though there are several MARL algorithms found in the literature, we have given the comparison of some well-known algorithms that employ the CTDE method to learn policies. The aim of this comparison is not to provide an exhaustive survey of MARL algorithms but to make a case for using a specific algorithm over the others for the proposed system model. Interested readers are referred to [42], a recent review paper of different MARL algorithms addressing scalability challenges. Moreover, we are focusing on DRL algorithms only.

Standard DQN [32] and actor-critic algorithms are extended to multiple agents using PS for homogeneous agents in [43]. This method scales well to a large number of agents

but it does not exploit any advantages of centralisation. Moreover, without using IDs of agents and without any cooperation between agents, we have observed in our simulations that these algorithms are not able provide better policies and they have convergence issues. However, this might be the issue for most of the centralized approaches. Our recent works [23] and [24] use the DQN with PS and in [24], we provided results for up to 500 devices for the bursty traffic. Similarly, just like the DQN can be extended to multiple agents using PS, one can also use Soft Actor Critic method [40] with a local actor and local critic that is shared among all users, where users' individual observation is used to update the actor and critic at each time step for all the users.

Other popular choices for MARL are multi-agent deep deterministic policy gradient (MADDPG) [33] and the recently proposed multi-agent proximal policy optimization (MAPPO) [34]. In both MADDPG and MAPPO, the critic network has a global view of the system, which is only applied during the training phase and actor networks are employed for each agent. An improved version of DDPG is proposed in [44] to exploit temporal correlations and improved computation cost. A major bottleneck of these algorithms is the scalability due to the shared critic network, even if the PS is considered. Since the shared critic network has the observation space of all the agents, the size of the observation space will grow exponentially with the number of agents. Moreover, in a network where the number of agents changes with time, it is inefficient to use the state-space of all agents at the centralized critic. counterfactual multi-agent policy gradients (COMA) [36] has a similar issue for scaling to a higher number of agents because a shared critic is used in it as well, just like MADDPG and MAPPO. In [45], an evolutionary algorithm is proposed that is useful for multi-task and multi-objective learning.

Some algorithms such as mean field MARL (MF-MARL) [38] and multi-actor-attention critic (MAAC) [39] show good scalability results but the major issue in these algorithms is that they require communication between the agents, which is not practical for our system model. Furthermore, the results for these algorithms are shown for a moderate number of agents.

VDN [37] and QMIX [35] are both for cooperative multi-agent learning in which joint action-values Q_{tot} are estimated from Q-values of individual agents that condition only on local observations. One of the main differences between these algorithms is that in VDN, the Q_{tot} is calculated as a linear combination of the Q-values of each agent, while QMIX employs a network that can compute Q_{tot} as complex non-linear combination of individual Q-values. This way of learning also provides better scalability as compared to MADDPG and MAPPO. QTRAN [46] improves upon both VDN and QMIX and provides more general form of factorization but it falls under the same category as VDN and QMIX. For these reasons, we will focus on the VDN and QMIX algorithms in our simulations.

VI. RL ENVIRONMENT AND MARL ALGORITHMS

A. THE ENVIRONMENT

We consider shared PRBs where each agent interacts with the resources by taking an action and receiving common feedback $F(k)$ as observation. Let $R(k) \in \mathbb{R}$ be the *immediate* reward that agents obtains at the end of time slot k after receiving the feedback. The reward is calculated for all the agents. In this work, the agents are assumed to be fully cooperative and hence they all share the global reward. For this reason, the subscript n is not used with $R(k)$. The action space of each agent is $\mathcal{A}$ and each device can either transmit on channel m, i.e., $A_n(k) = m$ or it can stay silent, i.e., $A_n(k) = 0$. The state of each device is the history tuple defined in (7) and (8) for IDs $= 0$ and IDs $= 1$ cases, respectively. The immediate reward depends on the agent n action $A_n(k)$ and other agents' actions $A_{n'}(k)$, $n' \neq n$. The accumulated discounted reward for an agent is defined as

$$\sum_{k'=0}^{\infty} \gamma^{k'} R(k + k' + 1), \tag{17}$$

where $\gamma \in [0, 1)$ is a discount factor.

Therefore, the reward function to maximize the packet success rate can then be defined as,

$$R(k) = \sum_{n=1}^{N} G_n(k), \quad \forall n \in \mathcal{N} \tag{18}$$

where $G_n(k)$ is calculated using (5).

The environment is assumed to be partially observable as each agent is unaware of the actions taken by the other users.

At each time slot k, each agent n obtains the feedback $F(k)$ from the receiver, updates its history and then feeds $S_n(k)$ to the proposed algorithm, whose output are the Q-values for all the available actions. Each agent n follows the policy π by drawing an action $A_n(k)$ from the following Boltzmann distribution,

$$\pi(a|s) = \frac{e^{Q(a,s)/\tau}}{\sum_{\tilde{a} \in \mathcal{A}} e^{Q(\tilde{a},s)/\tau}}, \qquad \forall a \in \mathcal{A}, \tag{19}$$

where $0 < \tau < \infty$ is the temperature parameter which is used for *exploration*. We decrease the value of τ to 0 gradually to make the agent more greedy.

B. DEEP Q-NETWORK (DQN)

DQN represents the action-value function using a neural network that are characterized by parameter θ. The target network is parameterized by θ^- that are periodically copied from θ during training. The DQN and its variants use a *replay buffer* to store the transitions (s, a, r, s'), where s is the actual state, s' is the next state that is observed after taking action a and receiving reward r. The learning updates are applied on the experience samples $(s, a, r, s') \sim U(\mathcal{D})$, that are drawn at random with uniform distribution as mini-batches of size z from $\mathcal{D}$ and by minimizing the following loss function,

$$L(\theta) = \sum_{i=1}^{z} \left[\left(y_i^{DQN} - Q(a_i, s_i; \theta) \right)^2 \right], \tag{20}$$

where $y_i^{DQN} = r_i + \gamma \max_{a_i'} Q(a_i', s_i'; \theta^-)$ is the target value for the i^{th} iteration. Since all agents use the same Q-network, we do not use subscript n.

The agents are trained using PS method, where they share the parameters of the same network, i.e., follow the same policy [43]. The policy is learned by using the experiences of all the agents simultaneously. However, as each agent receives different observations and the action selection is stochastic as in (19), this allows different behaviour between agents.

Since the environment is partially observable, the agents can benefit from using RNN such as gated recurrent unit (GRU) that can facilitate learning from previous history. A DQN making use of RNN is referred to as DRQN.

C. VALUE DECOMPOSITION NETWORKS (VDN)

VDN [37] take advantage of centralization and aim to learn the joint action value function $Q_{tot}(\boldsymbol{s}, \boldsymbol{a})$, a linear value decomposition from the team reward signal, where $\boldsymbol{s}$ is the joint observation of the agents and $\boldsymbol{a}$ is the joint action of agents. The VDN algorithm decomposes the Q_{tot} as the linear combination of the individual Q-values of each agent, i.e.,

$$Q_{tot}(\boldsymbol{a}, \boldsymbol{s}) \approx \sum_{n=1}^{N} Q(a_n, s_n; \theta), \tag{21}$$

Since we are using a single network for all the agents, we use θ without subscript n. The loss function of the VDN algorithm can be calculated in the same way as DQN, i.e.,

$$L(\theta) = \sum_{i=1}^{z} \left[\left(y_i^{tot} - Q_{tot}(\boldsymbol{a}_i, \boldsymbol{s}_i; \theta) \right)^2 \right], \tag{22}$$

where

$$y_i^{tot} = r_i + \gamma \max_{\boldsymbol{a}_i'} Q(\boldsymbol{a}_i', \boldsymbol{s}_i'; \theta^-) \tag{23}$$

is the target value for the iteration i.

In this way, each agent performs an action selection locally based on its own centrally learned Q-value in a decentralized manner. Moreover, the VDN method employs RNN or DRQN to calculate Q-values for each agent.

D. QMIX

The QMIX [35] algorithm improves the VDN and it can represent much richer class of action-value functions. QMIX applies the following constraint on the relationship of Q_{tot} and each individual action-value Q_n,

$$\frac{\partial Q_{tot}}{\partial Q(a_n, s_n)} \geq 0 \quad \forall n \in \mathcal{N}, \tag{24}$$

to ensure that mixing network has positive weights. Intuitively, it shows that if the weights of individual value function Q_a are negative, less weightage is given to that agent for cooperation. Moreover, as opposed to VDN, Q_{tot} is calculated in a complex non-linear way. QMIX uses a separate feed-forward neural network as a mixing network that takes individual agents' outputs and mixes them monotonically to produce Q_{tot} to enforce the constraint in (24) [35]. The weights of the mixing network are produced by a separate *hypernetwork* to ensure that they are non-negative. The loss function is calculated in the same way as given in (22).

TABLE 2. Simulation parameters.

Parameter	Value
τ	200 – 0.1
λ_n	0.3 (regular) and 0.015 (correlated)
Total Episodes	60
History size h	5 if RNN else 1
Learning rate	10^{-4}
Batch size	32
d_{th}	0.3
K^{τ}	500

E. COMPUTATIONAL COMPLEXITY

The only difference between the computational complexity of VDN and QMIX as compared to the DRQN is that VDN and QMIX require computing Q_{tot} during the training. If J is the number of layers in the neural network with U being the size of input layer, then the number of multiplications through the network is given by, $W_n = Ud_1 + \sum_{j=1}^{J-1} d_j d_{j+1}$ where d_j is the number of units in j^{th} layer. Therefore, the computational complexity for each device at each time step is given by $\mathcal{O}(W_n)$. Similarly, the computations of the network for N agents at each time step can be written as $W = \sum_{n=1}^{N} W_n + \tilde{W}$, where $\tilde{W}$ is the complexity of calculating Q_{tot} for the VDN and QMIX. Obviously, the calculation of Q_{tot} is not required and therefore $W = \sum_{n=1}^{N} W_n$ for DRQN. Therefore, the training complexity for E episodes, N agents and K time slots is given by $\mathcal{O}(EKW)$. The Q_{tot} computation by VDN is negligible as compared to the QMIX as it is a linear combination individual Q-values of the agents, whereas QMIX uses additional neural network to calculate Q_{tot}. For QMIX we can write the computational complexity $\tilde{W} = \tilde{U}\tilde{d}_1 + \tilde{d}_1\tilde{d}_2$, where we have used $\tilde{}$ to denote the corresponding notations of the mixer neural network in QMIX.

The complexity of the proposed MARL algorithms is high as compared to the EB schemes. However, the expensive computations are only required during the training phase at the central unit. For the scenarios of interest, each device follows a static policy that rarely requires an update.

VII. SIMULATION RESULTS AND DISCUSSION

In the following experiments, we use the VDN [37] method to learn RA policies for different values of N and M. We use the neural network with two layers of size 256 and a GRU layer of 64 neurons before the final layer of size M. The parameters used during the training of the network are presented in Table 2. The algorithm is given in Algorithm 1. The temperature parameter τ is updated after every K^{τ} time slots.

Algorithm 1 Training Phase of the Proposed Algorithm

Define $N, \tau, \gamma \in [0, 1]$, $\lambda_n, \forall n \in \mathcal{N}$, h and K
Initialize $S_n = 0, B_n = 0, \forall n \in \mathcal{N}$
  for each episode*do*
    for each time slot $k = 1, \ldots, K$*do*
      Generate traffic for all MTDs, i.e., $\tilde{B}_n \sim$ *Bernoulli*(λ_n) for regular traffic, and $\tilde{B}_n \sim$ *Bernoulli*(p) for ED traffic, $\forall n \in \mathcal{N}$
      Update buffer $B_n = \min(1, (\tilde{B}_n + B_n))$
      for each agent$n = 1, \ldots, N$*do*
        Observe input S_n and feed it to DRQN
        Generate the estimate of $Q(a_n, s_n; \theta)$, $\forall a_n \in \mathcal{A}$
        Choose action according to (19)
        Receive feedback $F(k)$ from the receiver
        Calculate $G_n(k)$ using (5)
        Update buffer B_n
        Observe the next state S_n
      end
      Compute reward $R(k)$ using (18)
      Store $(S(k), \mathbf{A}(k), R(k), S'(k))$ in the replay buffer $\mathcal{D}$,
      where $\mathbf{A}(k) = (a_1(k), \ldots, a_N(k))$
      Set $S(k) \leftarrow S'(k)$
      Sample a minibatch of size z
      for each transition i in minibatch, **do**
        Calculate y_i^{tot} using (23)
      Calculate loss $L(\theta)$ using (22)
      Apply gradient descent on $L(\theta)$, i.e., $\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta)$, to update Q-values
      if mixer = VDN:
        Calculate Q_{tot} using (21)
      elseif mixer = QMIX:
        Calculate Q_{tot} using QMIXer [35]
      for every K^θ time slots:
        Update $\theta^- \leftarrow \theta$
      for every K^τ time slots:
        Update τ
    end
  end

In all the experiments, we use experience replay to accumulate each agent's experience and the learning is performed with CTDE method. An agent's ID is one-hot encoded vector that is appended with the observation $S_n(k)$ of each agent. We will first provide results for regular traffic in which we also compare the results for different MARL algorithms such as QMIX and DRQN with VDN and then we present our results for the correlated traffic arrival model.

A. RESULTS FOR REGULAR TRAFFIC

For regular traffic, we employ two ways of training all algorithms: (i) using agents' IDs in the observation vector, which is a common way of training for CTDE, and (ii) without using them. We denote IDs = 0 as the case when agent IDs are not used and IDs = 1 as the case when we incorporate agent IDs in the observation space.

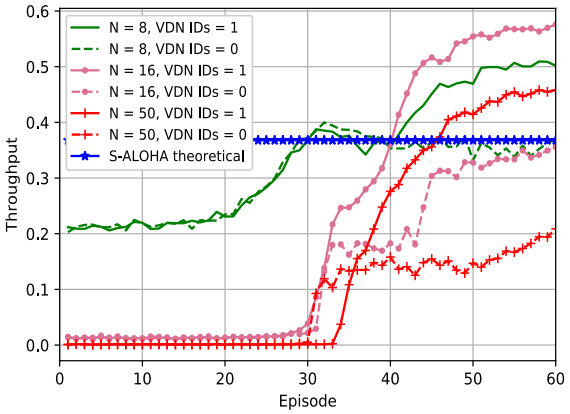


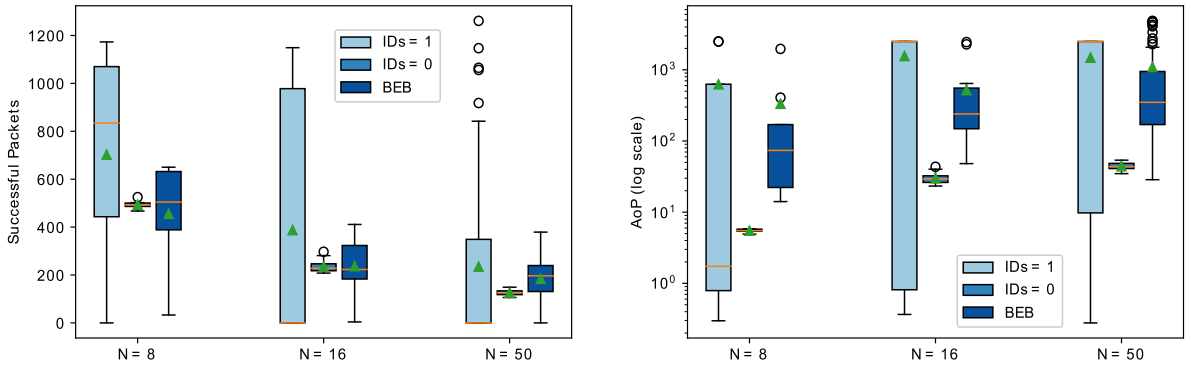
FIGURE 4. **Average throughput during training with VDN algorithm for different values of N and to compare the cases when using IDs and not using IDs. The results are for $\lambda_n = 0.3$ and $(K, N) = (2000, 8)$, $(K, N) = (3000, 16)$ and $(K, N) = (5000, 50)$.**

TABLE 3. **Average throughput and average AoP values for the proposed algorithm compared to the BEB for the learned policies shown in fig. 4**

		Av. Throughput			Av. AoP		
N	M	VDN IDs=1	VDN IDs=0	BEB	VDN IDs=1	VDN IDs=0	BEB
8	2	0.56	0.40	0.37	625.7	5.5	162.2
16	2	0.54	0.386	0.372	1480.4	29.8	519.3
50	5	0.44	0.25	0.36	1561.3	44.1	1095.1

We show the results in terms of average throughput (normalized reward) and AoP. The learning process and how average throughput of the system increases for different values of N is shown in Fig. 4. We used $K = 2000, 3000,$ and 5000 per episode during training for $N = 8, 16$ and 50, respectively. We increased K per episode as N grows to allow better learning for each value of N. Moreover, $M = 2$ is used to $N = 8$ and 16 and $M = 5$ is used when $N = 50$. The average throughput and average AoP after testing is shown in Table 3. Clearly, the case IDs = 1 outperforms BEB, slotted ALOHA theoretical throughput (i.e., $1/e$), and the case when IDs= 0. The case IDs = 0 provides much lower average throughput and as the number of devices N grows, the average throughput also decreases which is not surprising because agents decrease the transmit probability as N grows.

Moreover, we calculate whether the learned policy is fair or not in two different ways as depicted in Fig. 5. First, we show how many packets per user have been successful as in Fig. 5a and the second, the AoP of individual users (which shows both packet delay and fairness) as shown in Fig. 5b. We observe that using IDs incurs a significant unfairness



(a) Packets success distribution

(b) AoP distribution

FIGURE 5. Distribution of successful packets and AoP among MTDs for $N = \{8, 16, 50\}$ and $\lambda_n = 0.3$. We used VDN for both IDs = 1 and IDs = 0 cases. All cases tested for $K = 500$ time slots.

among devices as a subset of MTDs are starved out and they never get a chance to send a packet. Surprisingly, no IDs case provides much better fairness among devices. It is due to the inherent way the MARL algorithms with CTDE behave, we see that the case where IDs are omitted, provides us better fairness as compared to the other case. When we use IDs in the state space, devices are only concerned about achieving a better throughput. But when we do not use IDs, there's no such coordination that exists during the centralized training, which could allow the MTDs to come up with an unfair consensus.

To understand this, let us take the case of $N = 8$. For VDN, IDs $= 1$ case, it is clear that there are some devices that have sent 0 packets and there are a few devices that have sent most of the packets (95^{th} percentile is 1070 and 25^{th} percentile is around 443). Similarly, in Fig. 5b for $N = 8$, the average AoP value, which is the mean point and the distribution of AoP among each device shows a significant difference between 75^{th} percentile (627) and 25^{th} percentile (0.8). On the other hand, for VDN, IDs $= 0$ case, the number of successful packets sent by each device are around the mean value (493) for all percentiles, i.e., 75^{th} percentile is around 500 and 25^{th} percentile is around 486. Similarly, the average AoP value as shown Fig. 5b and in Table 3 for VDN ID$= 0$ case is much lower (5.5), which is the indication of fairness. Similar conclusion can be drawn for $N = 16$ and $N = 50$.

The case when IDs $= 0$ allows agents update the policies as if it is a single agent (hence single policy). This is unlike the IDs $= 1$ case in which, even if the state-space is the same for agents, they behave differently. This way we achieve better trade-off without using agent IDs, which is also scalable and allows devices to join/leave the network without identification. These plots also show that IDs $= 1$ case outperforms the BEB in terms of average throughput, but BEB has better average AoP than IDs $= 1$ case. The average throughput of BEB technique is higher than IDs $= 0$ case for higher values N but no IDs case exhibits much better throughput-fairness tradeoff, as evident from average AoP values. The case where agents IDs are used is most unfair because of the reward signal that only cares about maximizing the throughput.

Obviously, one can learn a policy by designing a reward function that enforces the devices to be fair even when agent IDs are incorporated; however, such a scenario is not of our interest in this paper.

1) COMPARISON BETWEEN VDN, QMIX AND DRQN

We used VDN algorithm to learn RA schemes for IDs and no IDs cases. In Fig. 6, we compare the performance of VDN with QMIX and DRQN for $N = 8$ and $M = 2$. Both VDN and QMIX use mixer networks to calculate the total Q-value Q_{tot} and exploit the benefits of centralized learning. However, DRQN does not take any such advantage of centralized training. For this reason, both QMIX and VDN algorithms learn a policy that maximizes the throughput for the case when agent IDs are incorporated (IDs $= 1$) and QMIX outperforms VDN. Interestingly, the DRQN learns only a slightly better policy when IDs $= 1$ as compared to the case when agent IDs $= 0$, again, due to the major difference that it doesn't take any advantage of centralization as opposed to the VDN and QMIX. However, in Fig. 7, QMIX and VDN clearly learn a policy that is unfair when IDs $= 1$, VDN being more unfair than QMIX. On the other hand, the DRQN for this case has lower AoP and relatively much fairer policy than VDN and QMIX. It does not imply that DRQN is a better algorithm than VDN and QMIX. In fact, QMIX outperforms VDN and both outperform DRQN as far as the objective (maximizing throughput) is concerned. Another interesting observation is that the learned policies are very similar between all the algorithms for IDs $= 0$ case and it is evident from both Fig. 6 and Fig. 7. They are fair but it seems that exploiting centralization advantages without using agent IDs does not provide significant improvements.

B. RESULTS FOR CORRELATED TRAFFIC

For correlated traffic arrival, we consider the example as shown in Fig. 3, in which $N = 20$ MTDs are uniform

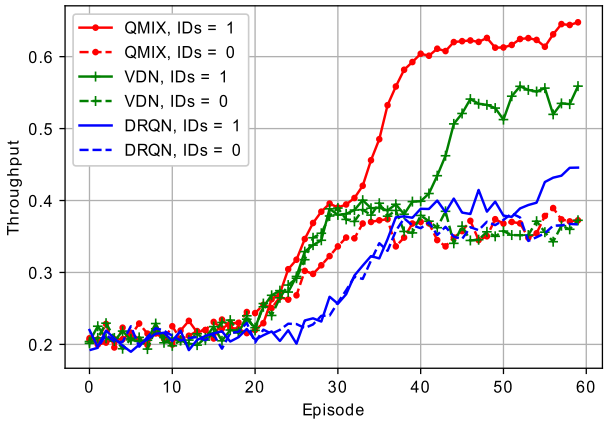


FIGURE 6. Average throughput comparison of different MARL algorithms during training, for $N = 8$ and $\lambda_n = 0.3$.

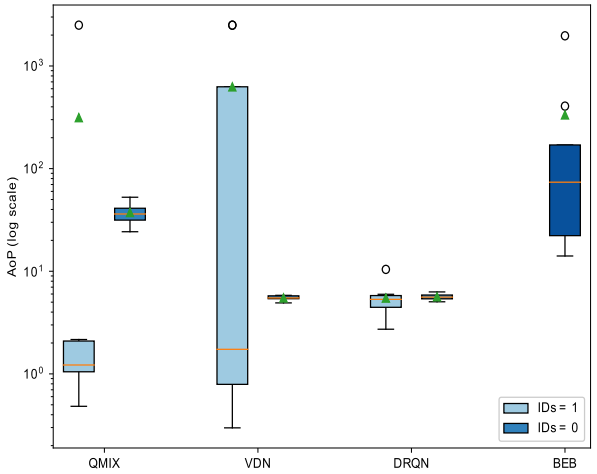


FIGURE 7. The AoP comparison among MTDs for different MARL algorithms for $N = 8$ and $\lambda_n = 0.3$.

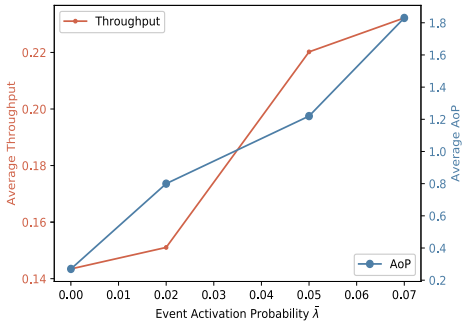


FIGURE 8. Average throughput and average AoP for different values of $\bar{\lambda}$ when $N = 20$, $L = 3$ and $\lambda = 0.3$.

randomly distributed in a rectangular area of unit size. The MTDs follow Bernoulli process with average arrival rate $\lambda = 0.3$ for regular traffic. We consider $L = 3$ event epicenters uniformly distributed at random. The MTDs belonging to each epicenter are given in Table 5. The events become active following another independent Bernoulli process with total average event activation probability given by $\bar{\lambda}$. If the probability of a given event is given by p then,

$$\bar{\lambda} = pL. \tag{25}$$

The overall average arrival rate of the system is then $\lambda + \bar{\lambda}$. Note that if a regular and ED packets both arrive at the same time slot in the device n's buffer, the regular traffic packet is dropped, and the priority is given to the ED packet.

The training and the testing is performed for different values of $\bar{\lambda}$ as shown in Table 4, with a fixed value of λ. The training for each $\bar{\lambda}$ is performed over 60 episodes and $K = 2000$ time slots per episode. Since are interested in designing an access policy for the MTDs deployed in an area, whenever an event l happens, the MTDs in the vicinity of the event or the MTDs closer to the epicenter of the event becomes active. For this, we calculate the probability of a device at location $\mathbf{x}$ becoming active due to the event happening at epicenter $\mathbf{y}$ as $p_{\mathbf{xy}}$ in the following way:

$$p_{\mathbf{xy}} = \begin{cases} 1 & \text{if } d_{\mathbf{xy}} \leq d_{th} \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

where $d_{\mathbf{xy}} = \|\mathbf{x} - \mathbf{y}\|$ and d_{th} is the threshold distance. We assume that the events are atomic in nature, i.e., if an event becomes active in time slot k, it activates MTDs within d_{th}. Each MTD activated by an event has one packet each to transmit and the MTDs remain active until their packets are successfully transmitted.

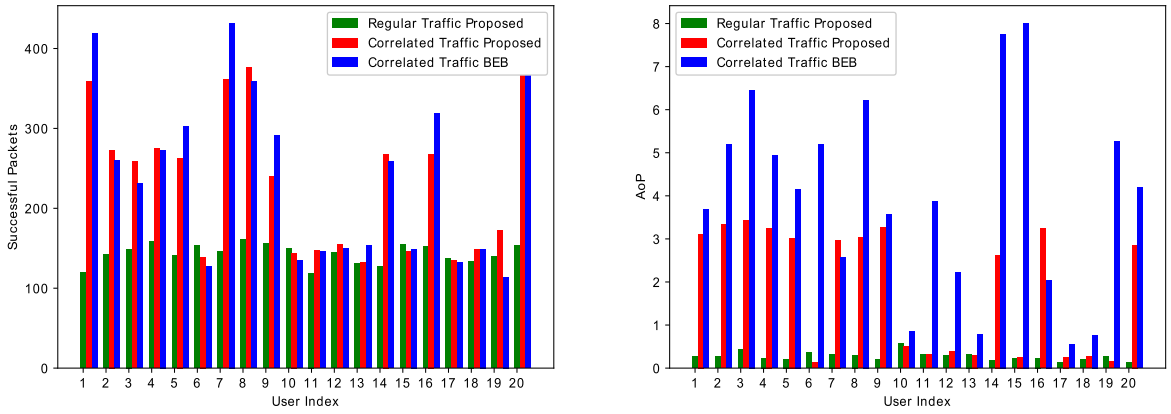
Since we want to learn the same policy for each agent, we do not use agent IDs for correlated traffic arrivals case and we only consider VDN IDs $= 0$ case since have seen that VDN performs better as compared to the QMIX and DRQN, Moreover, it is not considered that the ED traffic has any priority over regular traffic. All events are of the same nature and the same reward function as the regular traffic is used. Fig. 8 shows the average throughput and average AoP for different values of $\bar{\lambda}$. The value $\bar{\lambda} = 0$ means that there is only regular traffic and it can be seen in Fig. 8 that

TABLE 4. Number of times each event was activated for $K = 10,000$ time slots for different event activation probabilities $\bar{\lambda}$.

$\bar{\lambda}$	Event 1	Event 2	Event 3
0.02	63	61	59
0.05	170	170	169
0.07	243	226	241

TABLE 5. MTDs Reporting the events as in fig. 3.

Event #	MTDs Index
1	1, 7, 8, 16, 20
2	2, 3, 4, 5, 9, 14
3	1, 7, 8, 20



(a) Successful packets per user

(b) AoP per user

FIGURE 9. The comparison of successful packets per user and AoP of each user for ED traffic model for $(K, N) = (10000, 20)$, $L = 3$, $\lambda = 0.3$ and $\bar{\lambda} = 0.07$.

as the $\bar{\lambda}$ increases, the average throughput and the average AoP both increase, which is natural because when there is more traffic, there are more packets being successful but they require more time to be transmitted, hence the higher AoP. By further zooming in on individual MTDs, it can be observed how each MTD is behaving in correlated traffic scenario. In this case, agent IDs are not incorporated in the state-space of agents. Fig. 9 shows successful number of packets and AoP per device and the correlated traffic with regular traffic arrival scenario are compared. Clearly, only users that are involved in reporting any event have higher throughput as well as higher AoP. Obviously, when few users become active together, they will take more time to resolve collision and to send their packets successfully. The reason to show these plots is that the RL-based algorithms adapt to the traffic changes as the devices that are not involved in reporting any events have similar AoP and packet success rate to the regular traffic case, and only the MTDs belonging to events change their policies. The baseline BEB does not really adapt or care whether the devices are involved in events. The throughput is high for BEB for devices that are receiving more packets which is not surprising but in the AoP plot in Fig. 9b, there are devices that are not involved in reporting any events but they have higher AoP for BEB as opposed to the proposed algorithm.

C. SCALABILITY AND ROBUSTNESS ANALYSIS

We compare the learned policies of VDN, QMIX and DRQN for no IDs case and show how robust the policy learned is by each algorithm if it is scaled for a higher number of devices. The performance of each algorithm in terms of average throughput is shown in Fig. 10. We denote with N_{tr} the number of devices during training, and the number of devices for testing is denoted by N_{test}. The average arrival rate of the system is $\lambda = 0.3$. The results are simulated for 3 different random seeds and the best performances are shown in Fig. 10.

We show that the VDN has more robust policy than both QMIX and DRQN. For $\lambda = 0.3$, the policy learned for $N_{\text{tr}} = 4$ performs the same $N_{\text{test}} = 4, 8, 16$ and the throughput starts dropping after that. It is because the policy learned for $N_{\text{tr}} = 4$ has higher $\lambda_n = \lambda/N_{\text{tr}}$ and as the number of devices grow, the collisions are not resolved and hence the average throughput drops to almost 0. On the other hand, the policy learned for a relatively higher number of devices such as $N_{\text{tr}} = 16$ is robust for the number of devices less than N_{tr} and also scales for a large number of devices.

Intuitively, for instance, when $N_{\text{tr}} = 4$ and $\lambda = 0.3$, then λ_n for smaller N_{tr} has higher arrival rate or in other words, the MTDs observe packet arrival more frequently than λ_n for larger N_{tr} and therefore, devices learn to be more aggressive in terms of their transmissions to empty their buffers and such policy does not perform well for a very large number of devices.

Furthermore, the policy of QMIX has worse performance as compared to both VDN and DRQN. Moreover, QMIX without incorporating agent IDs is not as effective and not as robust as compared to the VDN.

D. ABLATION STUDY

The proposed algorithm considers only two parameters, i.e., the previous action $A_n(k-1)$ and the feedback signal $F(k)$ and their history of past h time slots is used to learn the access policy. The ablation study is performed to investigate the influence of the exclusion of past actions from the state space, which leaves only the broadcast feedback signal $F(k)$ for learning and it is shown in Fig. 11.

Clearly, including the past actions help to learn a better policy especially for ID$=1$ case. Agents can learn quicker and converge faster if the previous action is considered. The policy is not affected much with the exclusion of A_n for IDs$=0$ case and it because the agents are unable to cooperate effectively without knowing the identification and hence the policy for this case has convergence issues.

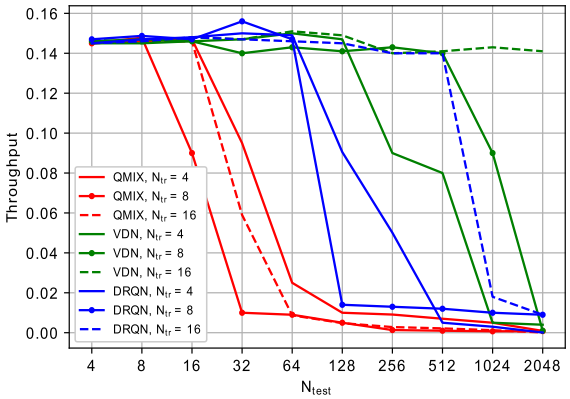


FIGURE 10. **Average Throughput performance of the learned policy for $\lambda = 0.3$ for different N_{tr} and tested for N_{test}.**

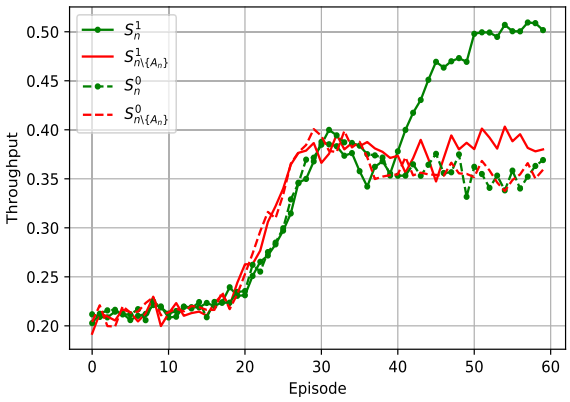


FIGURE 11. **Average throughput during VDN training when last action is not used in the state space S_n vs $S_{n\setminus\{A_n\}}$ for $N = 8$.**

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed MARL-based RA schemes for multi-user multi-channel mMTC network that use the broadcast feedback commonly received by all the MTDs to reduce signalling. We have shown that even when the optimization objective is to maximize throughput, not incorporating agent IDs provides a fair use of resources by each agent and, thus, results in a better throughput-fairness trade-off; not only when compared to the BEB scheme but also when IDs are used. This is supported by the analysis of successful packet distribution per user and the average AoP. We also show that incorporating agents IDs is not suitable for mMTC systems, as we aim to design a fair and a scalable scheme. Although using the agents' identification is useful for the policy convergence in MARL, the scalability analysis showed that the proposed algorithms (where agents IDs are omitted for learning) scales well for lower average arrival rates and that the learned policy is more robust for VDN as compared to the QMIX and DRQN.

Additionally, it is shown that RL-based algorithms can adapt to changes in traffic, whereas EB schemes are not made aware of such changes. The selected traffic model is representative of more complex scenarios with a correlated traffic arrival model along with the regular traffic. Results showed that the users learnt the correlation and adapted to the traffic variations.

Future works could consider prioritizing events to learn a scheme where the devices with high priority send their packets with low latency. For systems where the devices are fixed and known, the use of agent IDs could be further exploited to design a reward that is fair and that better exploits the correlation between devices. Other possible extensions may consider introducing certain degree of coordination among devices or group of devices to avoid throughput reduction when the number of devices increases.

REFERENCES

[1] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 74–80, Feb. 2014.

[2] C. Bockelmann et al., "Towards massive connectivity support for scalable mMTC communications in 5G networks," *IEEE Access*, vol. 6, pp. 28969–28992, 2018.

[3] C. Bockelmann et al., "Massive machine-type communications in 5G: Physical and MAC-layer solutions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 59–65, Sep. 2016.

[4] *Standards for the IoT*. Accessed: Sep. 20, 2022. [Online]. Available: https://www.3gpp.org/news-events/1805-iot_r14

[5] *Sigfox*. Accessed: Dec. 26, 2023. [Online]. Available: https://www.sigfox.com/

[6] *LoRaWAN*. Accessed: Dec. 26, 2023. [Online]. Available: https://lora-alliance.org/

[7] I. Leyva-Mayorga, C. Stefanovic, P. Popovski, V. Pla, and J. Martinez-Bauset, *Random Access for Machine-Type Communications*. Chichester, U.K.: Wiley, Dec. 2019.

[8] B.-J. Kwak, N.-O. Song, and L. E. Miller, "Performance analysis of exponential backoff," *IEEE/ACM Trans. Netw.*, vol. 13, no. 2, pp. 343–355, Apr. 2005.

[9] L. Barletta, F. Borgonovo, and I. Filippini, "The throughput and access delay of slotted-aloha with exponential backoff," *IEEE/ACM Trans. Netw.*, vol. 26, no. 1, pp. 451–464, Feb. 2018.

[10] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 310–323, Jan. 2019.

[11] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "Actor-critic deep reinforcement learning for dynamic multichannel access," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Nov. 2018, pp. 599–603.

[12] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.

[13] Y. Xu, J. Yu, W. C. Headley, and R. M. Buehrer, "Deep reinforcement learning for dynamic spectrum access in wireless networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 207–212.

[14] H. Li, "Multi-agent Q-learning for competitive spectrum access in cognitive radio systems," in *Proc. 5th IEEE Workshop Netw. Technol. Softw. Defined Radio Netw. (SDR)*, Jun. 2010, pp. 1–6.

[15] U. Challita, L. Dong, and W. Saad, "Proactive resource management for LTE in unlicensed spectrum: A deep learning perspective," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4674–4689, Jul. 2018.

[16] Y. Chu, S. Kosunalp, P. D. Mitchell, D. Grace, and T. Clarke, "Application of reinforcement learning to medium access control for wireless sensor networks," *Eng. Appl. Artif. Intell.*, vol. 46, pp. 23–32, Nov. 2015.

[17] L. de Alfaro, M. Zhang, and J. J. Garcia-Luna-Aceves, "Approaching fair collision-free channel access with slotted Aloha using collaborative policy-based reinforcement learning," in *Proc. IFIP Netw. Conf. (Netw.)*, Jun. 2020, pp. 262–270.

[18] S. Tomovic and I. Radusinovic, "A novel deep Q-learning method for dynamic spectrum access," in *Proc. 28th Telecommun. Forum (TELFOR)*, Nov. 2020, pp. 1–4.

[19] J. Kim, S. Kim, T. Taleb, and S. Choi, "RAPID: Contention resolution based random access using context ID for IoT," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 7121–7135, Jul. 2019.

[20] N. Jiang, Y. Deng, A. Nallanathan, and J. Yuan, "A decoupled learning strategy for massive access optimization in cellular IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 668–685, Mar. 2021.

[21] C. Zhang, X. Sun, W. Xia, J. Zhang, H. Zhu, and X. Wang, "Deep learning based double-contention random access for massive machine-type communication," *IEEE Trans. Wireless Commun.*, vol. 22, no. 3, pp. 1794–1807, Mar. 2023.

[22] A. Rech and S. Tomasin, "Coordinated random access for industrial IoT with correlated traffic by reinforcement-learning," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2021, pp. 1–6.

[23] M. A. Jadoon, A. Pastore, M. Navarro, and F. Perez-Cruz, "Deep reinforcement learning for random access in machine-type communication," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2022, pp. 2553–2558.

[24] M. A. Jadoon, A. Pastore, and M. Navarro, "Collision resolution with deep reinforcement learning for random access in machine-type communication," in *Proc. IEEE 95th Veh. Technol. Conf. (VTC-Spring)*, Jun. 2022, pp. 1–6.

[25] Y. Yu, T. Wang, and S. C. Liew, "Deep-reinforcement learning multiple access for heterogeneous wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1277–1290, Jun. 2019.

[26] Y. Hadjadj-Aoul and S. Ait-Chellouche, "Access control in NB-IoT networks: A deep reinforcement learning strategy," *Information*, vol. 11, no. 11, p. 541, Nov. 2020.

[27] Y. Shao et al., "Learning-based autonomous channel access in the presence of hidden terminals," *IEEE Trans. Mobile Comput.*, early access, doi: 10.1109/TMC.2023.3282790.

[28] A. E. Kalor, O. A. Hanna, and P. Popovski, "Random access schemes in wireless systems with correlated user activity," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[29] C. Zheng, M. Egan, L. Clavier, A. E. Kalør, and P. Popovski, "Stochastic resource allocation for outage minimization in random access with correlated activation," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2022, pp. 1635–1640.

[30] F. Moretto, A. Brighente, and S. Tomasin, "Greedy maximum-throughput grant-free random access for correlated IoT traffic," in *Proc. IEEE 94th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2021, pp. 1–5.

[31] H. Thomsen, C. N. Manchon, and B. H. Fleury, "A traffic model for machine-type communications using spatial point processes," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–6.

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

[33] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," 2017, *arXiv:1706.02275*.

[34] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. M. Bayen, and Y. Wu, "The surprising effectiveness of MAPPO in cooperative, multi-agent games," 2021, *arXiv:2103.01955*.

[35] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 21, pp. 1–51, Jun. 2022.

[36] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. AAAI*, vol. 32, no. 1. Palo Alto, CA, USA: AAAI Press, Apr. 2018, pp. 2974–2982.

[37] P. Sunehag et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proc. 17th Int. Conf. Auto. Agents MultiAgent Syst. (AAMAS)*. Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 2085–2087.

[38] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, J. Dy and A. Krause, Eds., Jul. 2018, pp. 5571–5580.

[39] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," 2018, *arXiv:1810.02912v2*.

[40] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018, *arXiv:1801.01290*.

[41] M. P. Mota, A. Valcarce, and J.-M. Gorce, "Scalable joint learning of wireless multiple-access policies and their signaling," in *Proc. IEEE 95th Veh. Technol. Conf. (VTC-Spring)*, Jun. 2022, pp. 1–5.

[42] T. Kravaris and G. A. Vouros, "Deep multiagent reinforcement learning methods addressing the scalability challenge," in *Multi-Agent Technologies and Machine Learning*, D. I. Sheremet, Ed. Rijeka, Croatia: IntechOpen, ch. 21, 2022.

[43] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems*, G. Sukthankar and J. A. Rodriguez-Aguilar, Eds. Cham, Switzerland: Springer, 2017, pp. 66–83.

[44] F. Song et al., "Evolutionary multi-objective reinforcement learning based trajectory control and task offloading in UAV-assisted mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 12, pp. 7387–7405, Dec. 2023.

[45] J. Chen, H. Xing, Z. Xiao, L. Xu, and T. Tao, "A DRL agent for jointly optimizing computation offloading and resource allocation in MEC," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17508–17524, Dec. 2021.

[46] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds., Jun. 2019, pp. 5887–5896.

MUHAMMAD AWAIS JADOON (Student Member, IEEE) received the B.Sc. degree in telecommunication engineering from the University of Engineering and Technology Peshawar, Pakistan, and the M.Sc. degree in electrical engineering from the University of Ulsan, South Korea. He is currently pursuing the Ph.D. degree with the Department of Signal Theory and Communications (TSC), Universitat Politècnica de Catalunya (UPC).

He is a Research Assistant with the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC) and an Early Stage Researcher (ESR) at ITN Windmill. As an ESR of ITN Windmill, he has had the opportunity to work at the Swiss Data Science Center, ETH Zürich, and Nokia Bell Labs, Paris, for his academic and industrial secondments, respectively. His current research interests include wireless communication and machine learning, with a focus on multi-agent reinforcement learning for resource management in massive machine-type communication.

ADRIANO PASTORE (Senior Member, IEEE) received the Diplôme degree from École Centrale Paris (ECP, now CentraleSupélec) in 2006, the Dipl.-Ing. degree in electrical engineering from the Technical University of Munich in 2009, and the Ph.D. degree from Universitat Politécnica de Catalunya in 2014.

From 2014 to 2016, he was a Post-Doctoral Researcher with the Laboratory for Information in Networked Systems (LINX), École Polytechnique Fédérale de Lausanne (EPFL), headed by Prof. Michael Gastpar. He is currently a Senior Researcher with the Research Unit on Information and Signal Processing for Intelligent Communications, Centre Tecnològic de Telecomunicacions de Catalunya. His research interests include information theory and signal processing for wireless communications, machine learning for communications, physical-layer network coding, protocol learning, quantum key distribution, and privacy–utility tradeoffs.

MONICA NAVARRO (Senior Member, IEEE) received the M.Sc. degree in telecommunications engineering from Universitat Politècnica de Catalunya (UPC) in 1997 and the Ph.D. degree in telecommunications engineering from the Institute for Telecommunications Research (ITR), University of South Australia, in 2002. From October 1997 to December 1998, she was a Research Assistant with the Department of Signal Theory and Communications, UPC, where she worked on the development of fractal shape multiband antennas for wireless cellular communications systems. She is currently a Senior Researcher with the Centre Tecnológic de Telecomunicacions de Catalunya, where she led the Communication Systems Division Group from 2016 to 2021 and is currently a part of the Direction Unit. She has also been a part-time Lecturer with Universitat Pompeu Fabra, Barcelona. Over the last 15 years, she has led projects funded by the European Commission, Spanish, and Catalan Governments, as well as the European Space Agency (ESA), spanning across 3G to 5G air interface designs, modem prototypes for space applications, virtualized wireless networks or intelligent transport systems. Her research interests include digital communications and information processing with applications to wireless communications and positioning. She served on the editorial board of *Emerging Telecommunications Technologies* (ETT).

ALVARO VALCARCE (Senior Member, IEEE) is currently the Head of the Department of Wireless AI/ML, Nokia Bell Labs, France. His research interests include the application of machine learning techniques to L2 and L3 wireless problems for the development of technologies beyond 5G. He is especially interested in the potential of multiagent reinforcement learning for emerging novel L2 signaling protocols, as well as in the usage of Bayesian optimization for RRM problems. His background is in cellular networks, computational electromagnetics, optimization algorithms, and machine learning.