University of Padova

Department of Mathematics

Master Thesis in Data Science

Reinforcement Learning algorithms for IoT communications over uncoordinated access channels

Supervisor
Leonardo Badia
University of Padova

Co-supervisor
Andrea Munari
DLR - Deutsches Zentrum für Luft- und Raumfahrt

Master Candidate
Chiara Cavalagli

Academic Year
2023-2024

Of course machines can't think as people do.
A machine is different from a person. Hence, they think differently.

The interesting question is,
just because something thinks differently from you,
does that mean it's not thinking?

Abstract

With the Internet of Things (IoTs), we intend any system of multiple devices able to collect and transfer information with any IT-device, like smartphones or computers, through the Internet connectivity. This thesis focuses on one of its applications which is massive Remote Monitoring Systems, where thousands of devices send time stamped updates over a wireless channel to a common receiver. We focus on the underlying - link layer - communication protocol of such scenario. A simple yet powerful random access protocol, the slotted-ALOHA, is used as benchmark model to organize the channel access strategy. The overall objective to keep an up-to-date perception at the receiver in terms of the freshness of data, described by the Age of Information (AoI), a novel metric which tracks the timeliness of a specific process with respect to the receiver. Under the paradigm of Reinforcement Learning, the proposed algorithm - AoI-Q-ALOHA method - tackles a Multi-Agent setting in a decentralized fashion, where each user is independently learning an AoI-based channel access policy. To let the agents adapt to a shared channel, a binary success/collision feedback distributed by the receiver is used as the main global incentive during the learning process. Without any assumption on the channel population, team behaviour is encouraged through a tailored individual reward designation, based on the current value of AoI. We compare the performance in terms of the mean channel AoI, Throughput and Fairness index, to the standard slotted-ALOHA protocol, and to the threshold-ALOHA policy, a benchmark protocol that resorts to a central optimization of the access parameters. Interesting insights on the distributed setup are derived, as well an exhaustive survey on the properties and robustness of the algorithm.

Contents

Listing of figures

1 Introduction

With the Internet of Things (IoTs) [1], we intend any system of multiple devices able to collect and transfer information with any IT-device, like smartphones or computers, through the Internet connectivity. The integration of such a communication medium led many daily activities to be easily accessed remotely, without the human intervention in place. From industrial areas to health care and smart cities/homes, the IoTs is now shaping the way we live.
This thesis focuses on one of the IoTs applications: massive Remote Monitoring Systems [2], a network of thousands of users each of them tracking a specific process, usually via the usage of sensors, with the goal to update the main receiver over time. Commonly applied in smart homes [3], as well as many other industrial tasks [4], daily checks like temperature monitoring, fault detection, security and so on, are now remotely tracked without the need to directly intervene in place.
We focus on the underlying - link layer - communication protocol of such scenario. A simple yet powerful random access protocol, the slotted-ALOHA (SA) [5], is used as benchmark model to organize the channel access strategy. Within a slotted-wise horizon of events, each user is able to update the main receiver through the transmission of data packets. Each user is slot-synchronized, and we consider the packets duration being equal to a slot. If two or more users transmit a packet in the same slot, a collision occurs and the data is considered to be lost. A packet is said to be successfully detected by the receiver if it is the only one transmitted in a given slot. We do not consider re-transmission of lost packets. The random access protocol notation refers to the channel access strategy: each user attempts a transmission with a fixed

probability, which is usually described under the assumption of a Poisson distributed arrival rate. SA is well-known for its best-case scenario performances in terms of throughput [6]- rate of successful transmissions over the total attempts - being equal to the 36% when the channel load is at $1\frac{pkt}{slot}$. However, when it comes to tracking systems, the throughput no longer becomes one of the main concerns as the main goal is to keep the central receiver up-to-date. In fact, a novel metric, the Age of Information (AoI) [7], has recently gained attention thanks to its focus on the timeliness of a specific process with respect to the receiver. For each slot t and for a generic user, the AoI is formulated as follows:

$$\delta(t) = t - \sigma(t),$$

where $\sigma(t)$ is the latest slot in which the node has successfully informed the receiver. The AoI is a linear function that grows with time when there are no transmissions or collisions; resets to 1 otherwise. Similarly for the throughput, the standard performances of a SA within the best case scenario ($1\frac{pkt}{slot}$ of channel load, 36% of throughput) brings an average channel AoI of $\approx 2.7\frac{slots}{nodes}$, an output value that, if moved to a massive network scenario, arises concerns on the channel timeliness. Thus, the objective of this thesis focuses on the minimization of the channel AoI. Moreover, we remark that by keeping the channel freshly updated as possible, we intend to frequently reset the age of each node, and so the attention is shifted to the system throughput as well.

In order for the nodes to develop ad-hoc channel access policies, the paradigm of Reinforcement Learning (RL) has been chosen. Considering the need to implement real-time services, RL comes to help, especially when solving online-tasks within dynamic environments and by learning solely from raw experience. Here, to keep the framework reliable as possible to real-world scenarios, the set up has been kept *decentralized*: each node is solely aware of the presence of the receiver; we do not consider any communication among the users. In order for each user/*agent* to be able to track its own AoI over time, we consider a feedback-based channel model, where the receiver sends at the end of each slot an acknowledgment signal (ACK) to every user containing the outcome of the latest channel contention i.e. $\{-1$ (coll), 1 (succ)$\}$ but without mentioning which users had attempted the transmissions.

As in every RL-based algorithm, the definition of the states, actions and learning paradigm are the keys for the solution. Considering a set of actions for each node i.e. $\{wait, trans\}$, and the state being assigned by the current age $\delta(t)$, the problem is simple enough to be tackled by a tabular approach like Q-learning, a fast and versatile off-policy algorithm. Considering the

deterministic nature of such a method, plugged into a probabilistic problem like SA, an integration of an adaptive transmission probability will be considered whenever the agent will choose to transmit. Moreover, in a Multi-Agent decentralized fashion, many challenges arise. The unavailability of sharing information among the agents, and the lack of cooperativeness and competitiveness among the users, brings to the absence of convergence guarantees. Nonetheless, as it will be shown, team behavior is encouraged through a tailored individual reward designation. A significant improvement in terms of AoI will be displayed.
In the next Chapters, a background introduction to the problem is given, to then continue with the framework formulation as well as the Algorithm description. Related results and discussions will follow.

2 Background

2.1 Wireless Sensor Monitoring for IoTs

We refer as Wireless Sensor Networks (WSNs) [2] to a system in which several devices are monitoring a specific process, typically a physical condition of the surrounding environment, to collect and forward their updates to a common receiver. As the name suggests, each network user exchanges data through wireless connectivity, which often occurs via solutions such as Wi-Fi or Bluetooth.
A broader idea that emerged alongside WSNs is the Internet of Things (IoTs). This type of technology includes several activities related to process monitoring, but goes beyond that: the range of applications varies from the industrial field to the home automation (smart homes), medical care and so on. With the rise of IoTs, a wide variety of objects and devices can now communicate between each other and, like the case of our interest, get monitored remotely without the physical intervention of humans. Here, the Internet connectivity will serve as a mean to update all of the IT-devices connected to the main receiver. Because IoTs systems comprehend a variety of applications, there are no rules on the usage of the underlying communication protocol. In our work we will focus on Remote Monitoring Systems which will be modelled on the link layer by a simple yet powerful protocol, the Slotted ALOHA (SA), within a massive multi-user scenario. In the following sections, we will delve into the system model, relevant performance metrics, and the challenges that motivated us to explore its improvements.

2.1.1 Slotted ALOHA

The ALOHA protocol had its first implementation back in the $1970s$ at the Hawaii University to provide radio communications within the archipelago [5]. We will focus on one of its famous variants, *slotted* ALOHA, where the time space is divided into fixed-size portions that we will call slots.

Specifically, let us consider a network of n devices, each of them communicating with a common sink $\mathcal{S}$ through the exchange of packets of data of size p. Each user is slot-synchronized, and we will consider from now on each packet having the size of a slot, so that the transmission will last only one slot.

If two or more users send their data in the same slot t, then a collision occur, and the information is considered to be lost due to interference. Generally speaking, when a collision occurs, ALOHA protocols allow re-transmissions. However, in our case we do not consider re-transmission of collided packets. A packet is said to be successfully decoded by the sink if it is the only sent over the channel, at a certain time slot t. It is possible then to describe the channel outcome at every slot as one of the following conditions: $\{$*reception, collision, idle*$\}$. These assumptions are commonly referred to as collision channel model [8], and are broadly used to model the performance of random access protocol.

The slotted setting was brought up from the necessity to reduce the vulnerable time of collisions among packets. In a pure ALOHA network, where time is continuous, a collision is avoided if, fixed a transmission at time t, no users have accessed the channel in the time frame of $[t - p, t + p]$, where p is the packet size. Within a slotted assumption, the vulnerable time is halved as shown in Figure 2.1. Without loss of generality, we will consider from now the slot size, and so the packet size, being unitary.

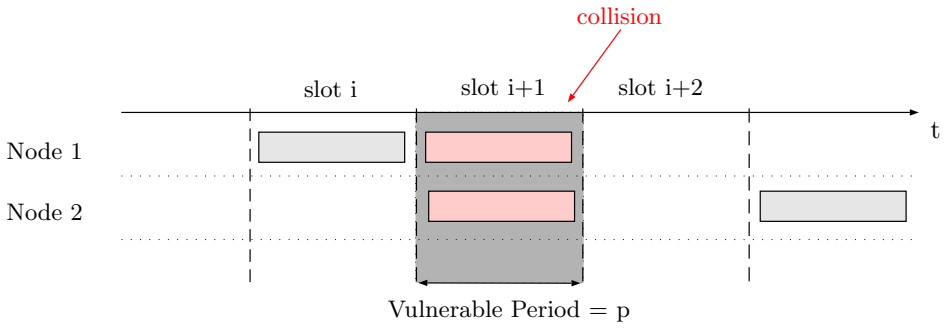

Figure 2.1: Example of a collision in a two-users slotted ALOHA network. The vulnerable period p is equal to the slot size.

The slotted ALOHA (SA) is a common example of how to operate a random access channel, whose name indicates the type of medium access strategy that is chosen by the users. Here, each node sends a packet in the first slot available as soon as it produces one. Regardless of the activity of the other devices, the channel contention is strictly related to the arrival rate of the packets, that we assume being equal for all users and Poisson distributed. Accordingly, by denoting with $\mathcal{Z}$ the number of packets sent in an infinite time horizon, the following statement holds: *

$$\mathbb{P}[Z = z] = e^{-\lambda}\frac{\lambda^z}{z!} \tag{2.1}$$

Because we are interested into expected values, we will refer from now to $\mathbb{E}[Z] = \lambda = G$ as the *traffic* (G) of the channel.
In general, an estimation of the load can be achieved empirically, as displayed below:

$$G = \frac{\#trans}{T} \in [0, \infty[, \tag{2.2}$$

where $\#trans$ is the number of transmissions in the channel, T is the time horizon. The equation is an empirical approximation of the normalized traffic. Moreover, under the assumption of ergodicity and a large enough network:

$$G = \frac{\#trans}{T} \underset{n\to\infty}{\backsimeq} \lambda$$

Considering n devices having the same transmission probability ρ, it follows that for every value of traffic λ, the single-user access probability is equal to $\frac{\lambda}{n}$.
In conclusion, the Poisson distribution describes the general load of a random access protocol. However, different arrival rates and access strategies can be implemented. From now on, we assume the model under the assumption of "generated-at-will", where at each slot, a node has a new packet to send.
Given the tools to understand a SA protocol, we can now analyze its well-known results in the following section.

*For a large network size ($n \to \infty$), the number of packets arrivals in each slot can be approximated by the Poisson distribution. As the size grows, the mean λ remains finite and constant. Thus, such approximation is appropriate only for a largely enough network [9].

Performance of SA - Throughput, Packet Loss Rate, Fairness Index

When talking about network traffic, we intend every single transmission that has been initialized so far; this clearly includes successfully delivers but also collisions. Recalling that a collision makes the transmitters lose their packets, we want to quantify the actual rate of decoding of the receiver. To this simulation, we consider the throughput S ([6]), which computes the average number of successful transmissions over a given time horizon T. Let us show the main result related to SA.

Results 2.1.1 (SA throughput)**.** Given a network of n devices, configured in a slotted-fashion protocol, with a Poisson arrival rate of packets λ, we have that the system throughput is given by the following equation:

$$S = Ge^{-G}, \tag{2.3}$$

where G is the traffic of the network.

The derivation of 2.3 can be easily obtained from the assumption of having a Poisson-distributed traffic, combined with the requirement of having a single transmission per slot (i.e. $\mathbb{P}[\mathcal{Z} = 1]$). The empirical approximation of the network throughput after T slots is computed as follows:

$$S = \frac{\#succ}{T} \in [0, 1], \tag{2.4}$$

where $\#succ$ is the number of successful transmissions so far, and T is the time horizon. The

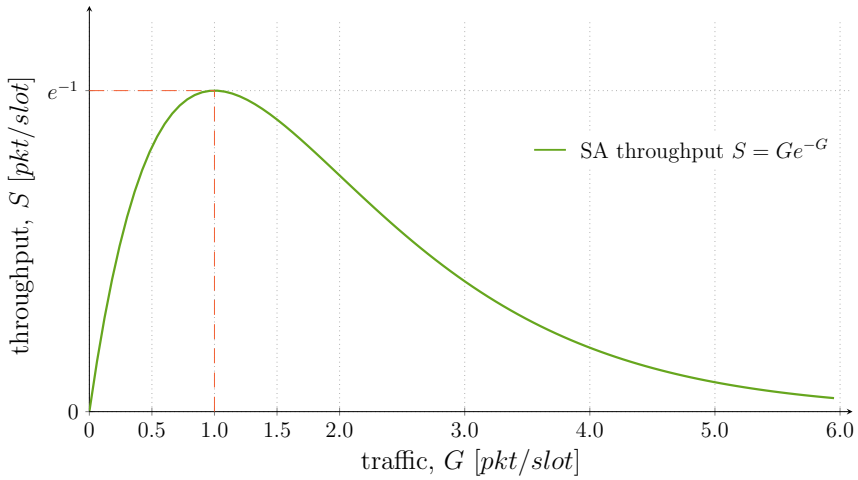

Figure 2.2: Throughput system versus channel traffic.

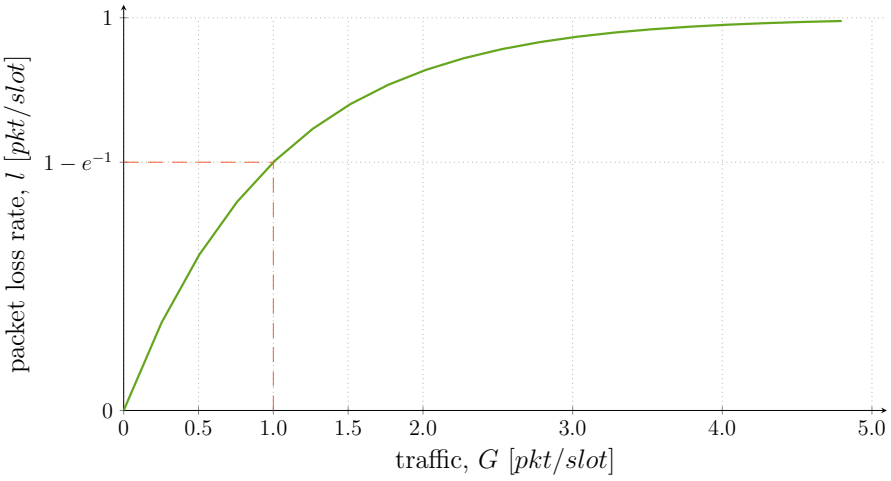

Figure 2.3: Packet loss rate versus channel traffic.

analysis of the best performances of SA is straightforward, and is shown in Figure 2.2. The peak of the throughput is at $e^{-1} \approx 0.36$ when the average traffic load is 1 packets per slot.

Alongside the throughput, there is the *Packet Loss Rate* $\mathcal{L}$ that, as the name suggests, describes the probability for a sent packet to not be decoded, due to collisions. Its formulation is straightforward:

$$\mathcal{L} = 1 - \frac{S}{G} \overset{(2.3)}{=} 1 - e^{-G} \in [0, 1] \tag{2.5}$$

From Figure 2.3, we can see how the function asymptotically reaches 1 (i.e. total loss of packets) as the channel congestion increases. It is monotonically increasing, thus leaving little room for the optimal traffic loads choice. That means that in the best case scenario of a load of $1 \frac{pkt}{slot}$, users will still have a very low chance of having their packets collected by the receiver; moving into the context of interest that is massive IoT systems, those results are not promising.

Another metric that will be used to evaluate the channel is the *Fairness Index*, a value which indicates the quality of the bandwidth share. The goal is to equally allocate the successful transmissions among the users. Such thing is naturally happening in a fixed configuration where the transmission probability τ is equal for each node. In a RL-based framework, where the access strategy changes over time and over different users, the fairness index help to understand the resource allocation of the running algorithm. There are several formulas in literature that can be used; the Jain's fairness index is the chosen benchmark for this thesis. Its formulation is the

following:

$$\mathcal{J} = \frac{(\sum_{i=1}^{n} s_i)^2}{n \cdot \sum_{i=1}^{n} s_i^2}, \tag{2.6}$$

where n is the number of devices and s_i is the throughput rate of the i-th node.
In conclusion, throughput S, Packet Loss Rate $\mathcal{L}$ and Fairness index broadly describe performances of SA. We shall recall the configuration to be quite standard: no interaction between the users and the central receiver, no changes in terms of transmission probabilities, and no inferences about the channel statistics. In the next section, we propose a novel metric that adds awareness to the devices in order to adapt their channel access strategy for the sake of improvements.

2.1.2 Information Freshness

When talking about monitoring systems, keeping the central receiver up-to-date becomes one of the main concerns. Anomalies, emergencies or critical phases need to be promptly managed. Focusing only on the optimization of the successful rate may not meet the expectations of the application of interest. In some cases, few instants of delay can cause irreversible damage. The notion of *freshness* of data became inevitably relevant. As the real-time services in IoT demand was constantly evolving, a novel metric has emerged along the way [7].
The Age of Information (AoI) refers to the receiver's freshness knowledge of a specific process, monitored by a channel user i, and it measures the time elapsed between the latest successfully decoded packet from the source and the current time stamp t. Its formulation is the following:

$$\forall i,\ \delta_i(t) \coloneqq t - \sigma_i(t) \tag{2.7}$$

where i is the user index, and $\sigma_i(t)$ denotes the latest instant in which the i-th user has informed the sink about its state, at current time t. We shall recall that the time unit t is represented in our case by a single slot, but unslotted models have been subject to the AoI-analysis as well, [10].
The function grows linearly with time if the node does not sends any new updates or collides, resets to 1 otherwise. Indeed, if a packet has been successfully decoded, then the elapsed time is just the length of the slot i.e. $\sigma(t) \leftarrow (t+1)$ and so the age becomes $\delta(t) = 1$. An example of an age profile of a node is showing in Figure 2.4 .
In random access protocols like SA, each node is able to track its own age only if they are aware of the outcome of the latest slot contention. Such things can be possible thanks to an acknowledgment (ACK) signal: a binary outcome distributed by the receiver among the active nodes

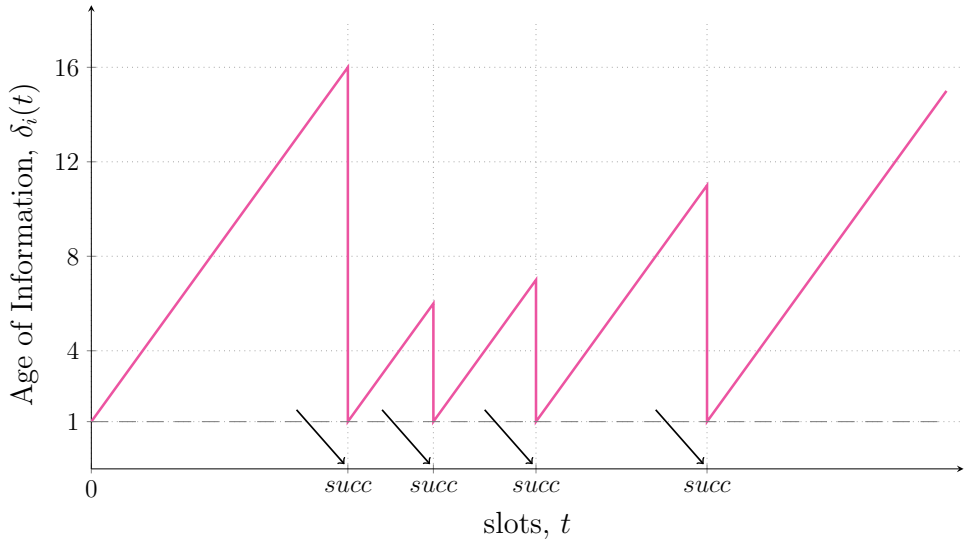

Figure 2.4: Age of Information of a node. When it successfully transmit a packet ($succ$), the age resets to 1.

of the channel. The latter is a type of *immediate feedback* $\mathcal{F}(t)$ that comes at low costs in massive-monitoring systems. Its formulation is the following:

$$\forall\, slots\, t,\ \mathcal{F}(t) \in \{-1(collision),\ 1(success)\} \tag{2.8}$$

Similarly to the throughput and traffic, we shall focus from now on the normalized average AoI of the network:

$$\overline{\Delta}(n) = \frac{1}{n}\sum_{i=0}^{n-1}\Delta_i, \tag{2.9}$$

where n is the total number of users, and Δ_i is the average age of the i-th user over a time horizon T ([11]):

$$\Delta_i = \frac{1}{T}\sum_{t=1}^{T}\delta_i(t). \tag{2.10}$$

It is trivial to observe that the latter will be computed in a discrete-manner over an horizon-time T large enough to obtain reliable values. It has been shown how the mean AoI is linked up to the throughput in a "generate-at-will" [12] SA-like protocol, where each node is considered to have a fresh packet to send per each slot, via the derivation of the equation ([11]):

$$\overline{\Delta}(n) = \frac{1}{2} + \frac{n}{S}, \tag{2.11}$$

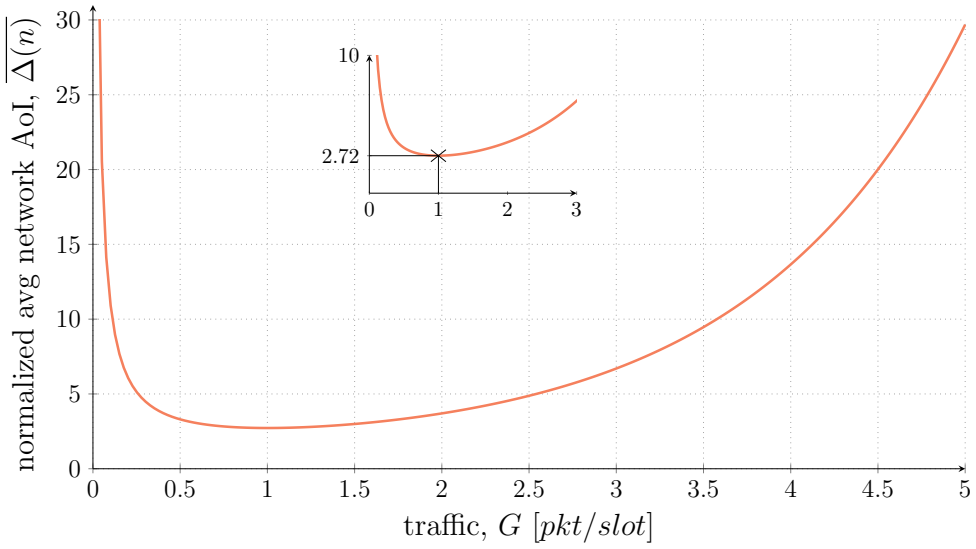

Figure 2.5: Normalized mean channel AoI against fixed traffic loads.

The relation shows how AoI and throughput are tightly linked up: minimizing the age may leads to maximizing the throughput. Therefore, optimizing the channel in terms of AoI, directly shift the attention to the throughput.
On the other hand, improving the system throughput does not necessarily improves the channel mean AoI. Increasing the transmission rate for individual users might seem beneficial, raising the chances of a successful detection. However, as the network size grows (i.e. the number of devices), the collision probability increases, leading to less frequent age resets for the users.

As we did it for the throughput, we can show the standard performances of a SA protocol for different traffic loads in terms of mean AoI. By looking at Figure 2.5 that is showing the normalized mean age of the channel (equation 2.9), we can state that the age is indeed linked with the throughput as we can get the minimum of $2.72 \frac{slot}{node}$ at the corresponding load value of $1 \frac{pkt}{slot}$, when the throughput reaches the 36% of success rate. Thus, in the best case scenario, the receiver will wait *at least* 2.72 slots to get a *fresh* update from one of the channel users. Even if it may seem like a remarkable result, we have to recall that the latter are intended as normalized values over the number of devices. This means that every node will have to wait on average $2.72 \times n$ slots to successfully deliver the updates. Moving to massive IoTs system scenarios, it is clear that the monitoring performances degrade as the number of nodes increases. To improve that, a variety of policies for the users have been proposed in literature. We now present a threshold-based strategy that inspired the making of this thesis.

AoI-aware Policies: threshold-ALOHA

Minimizing the average channel age certainly requires a level of cooperation between the users. Indeed, one user could choose to lower its mean age without taking into consideration the others, and that may lead to an increase of the overall channel age because of the unfair share of the bandwidth. It is then necessary to state when is the right time for an user to transmit.
In the work presented in [12], the authors propose a threshold-based policy where each user remains silent when its age is below a fixed threshold Γ, and transmits following SA otherwise. Such strategy categorizes the users between active and inactive, depending on their age being below or above Γ. The latter is set so that every node gets the chance of resetting the age in a reasonable amount of slots, while the fresher nodes are being silent.
The authors provide an asymptotic formula for the optimal parameters of the threshold-based policy. Under the optimal configuration, the network shows remarkable results in terms of AoI, at the cost of losing a small percentage of the system load and the throughput as well. Specifically, the optimal parameters such as the threshold, transmission probability and mean AoI were obtained as:

$$\lim_{n\to\infty} \frac{\Gamma^\star}{n} = 2.17 \tag{2.12}$$

for the threshold,

$$\lim_{n\to\infty} n\tau^\star = 4.43 \tag{2.13}$$

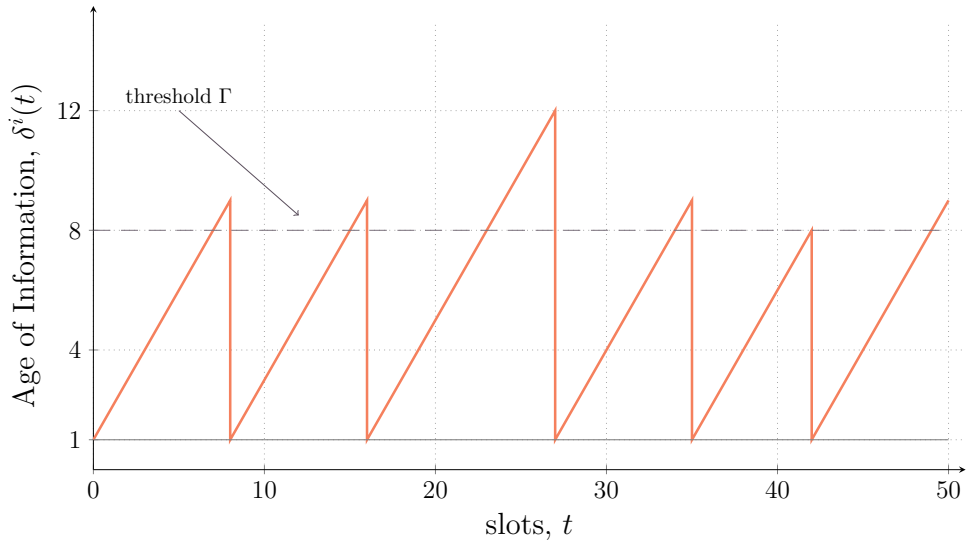

Figure 2.6: Age profile of an user under the threshold policy.

for the transmission probability; and lastly:

$$\lim_{n\to\infty} \frac{\overline{\Delta}^{\star}}{n} = 1.42. \tag{2.14}$$

for the mean age.
Such results are asymptotic, and optimized for a very large network population channel. On the other hand, no analytical results for the optimal performance for lower values of n were available. To study this aspect, we have performed an accurate exploration (via simulations) of the (Γ, τ) space, where τ is the transmission probability and Γ the threshold. A visible discrepancy between the theoretical and empirical results for small networks (i.e. around thousands of nodes) are showing in Figure 2.8.

For a small network of 100 nodes, the normalized average age of the channel per different thresholds and per different transmission probabilities is reported in Figure 2.7. In all of the presented curves, a very steep increase of the overall age is present after a certain transmission probability. To explain such pattern, we shall remember that by keeping inactive the freshest nodes, the network will have the number of active nodes (i.e. above the threshold) concentrated around a certain value. Such quantity, that empirically concentrates around the 20% of the totality of users, behaves like a normal SA under a smaller topology configuration. Thus, the sub-network of active nodes will show standard SA performances if the transmission probability is properly set. If the latter is set too much high, then the channel would be congested

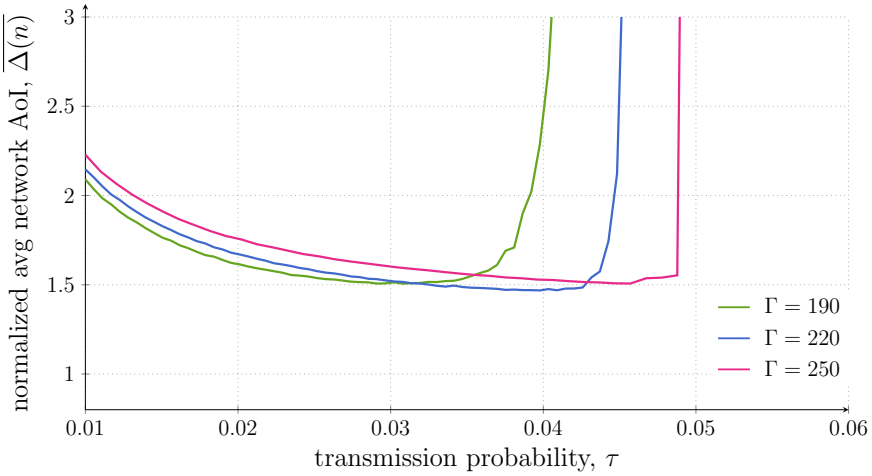

Figure 2.7: Normalized average network age per different fixed thresholds, against the transmission probability. The network size is set to 100 nodes.

	Nodes	$\Gamma^\star$	$\tau^\star$	$\overline{\Delta}^\star/n$
Sim	100	215	0.04	1.47
Asympt	100	220	0.04	1.41

Table 2.1: Comparison between asymptotic and simulated results for a network size of 100 nodes.

and so the system throughput and mean AoI would degenerate. Moreover, setting the threshold excessively large (250 in example shown in Figure 2.7), may lead to an high sensitivity to small changes in terms of transmission probability due to the high number of inactive nodes that will start to contend the channel once their age has been reached.

The optimal setting of the toy example compared with the theoretical one is showing in the Table 2.1. As expected, the results fluctuate around the theoretical ones, that we know being more accurate as the number of nodes increases. Such trend has been confirmed via dedicated simulations, whose outcome are reported in Figure 2.8.

We now compare the obtained results for a network of 100 nodes, with the standard SA ones, in order to highlight the losses and wins of the proposed policy. From the Table 2.2 we can acknowledge that the mean age is significantly decreased, at the cost of losing a negligible portion of traffic ($\approx 0.90 \frac{pkt}{slot}$ instead of the standard $1 \frac{pkt}{slot}$). Other metrics like throughput and index of fairness are preserved. That is clearly an improvement of the SA performance,

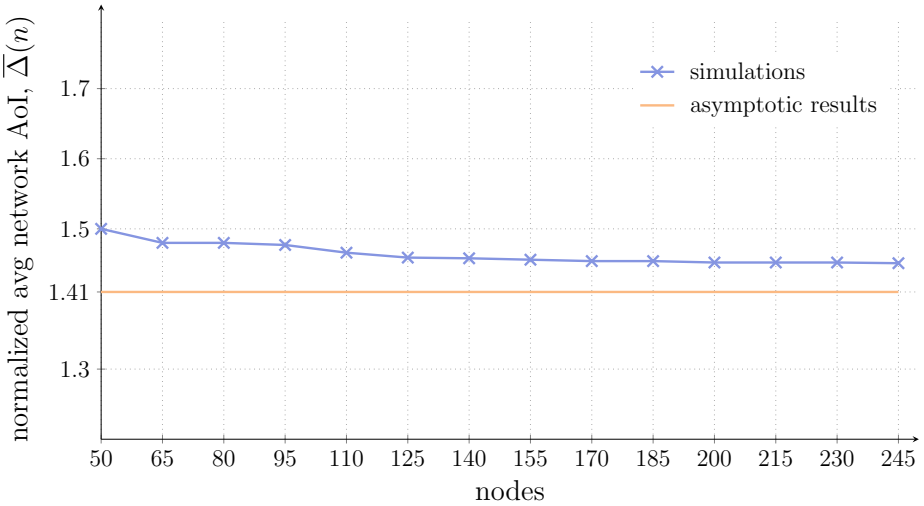

Figure 2.8: Normalized mean age against the size of the network.

Table 2.2: Comparison between SA and threshold-ALOHA performances.

	Slotted ALOHA	threshold-ALOHA
Normalized mean channel AoI, $\overline{\Delta}$	2.72	1.47
Traffic, G	1	0.90
Throughput, S	0.368	0.365
Fairness, $\mathcal{J}$	0.99	0.99

especially when dealing with massive multi-users networks. However, as discussed, retrieving the optimal hyper-parameters requires the knowledge a priori of the number of nodes. Shifting to a monitoring system scenario, channel insights such as the number of devices may be unknown, and finding the proper transmission probability within a dynamic network may be challenging and costly. Therefore, this thesis aims to improve a SA protocol in terms of AoI inspired by a threshold-based policy, without leveraging the knowledge of network cardinality and other relevant features such as the transmission probability.
In order for us to implement ad-hoc strategies for the users, Reinforcement Learning has been chosen thanks to its versatility in dynamic processes. In the following sections we introduce the general framework to then deepen into the algorithm chosen to approach the problem.

2.2 Reinforcement Learning: an introduction

Automation and prediction processes have been shaped by Artificial Intelligence in many industries and scientific fields. Among the most known paradigms, Machine Learning has the primacy on teaching how to perform several tasks without being explicitly programmed to. The continuous demand has led to the birth of various research areas. Our interest falls under the name of optimal control, a research field focusing on the development of strategies for sequential-decision making systems, within dynamic environments. Such area has experienced significant growth around the start of the 20th century, thanks to the rise of Reinforcement Learning. Originally proposed to solve the well known k-armed bandit problem, Richard Bellman [13] extended such scenarios from immediate returns to the inclusion of long term interactions, leading to a plethora of algorithms that are currently the state of the art of many application fields [14].

Reinforcement Learning (RL) is a branch of Machine Learning, that distinguish itself mainly on how information is treated and processed. Thinking about a classification scenario, the model has to learn how to correctly predict a sample. The pattern is the same among different target-given problems: throughout the whole training, the model knows if the output is correct or wrong, independently on how it chose the outcome. This type of feedback is called *instructive feedback*, and it returns an evaluation of the decision regardless of the action taken, [15]. On the other hand, the *evaluative feedback*, takes into account the action chosen from the model and it gives a rate on how good it is, without mentioning if it is the best or the worst

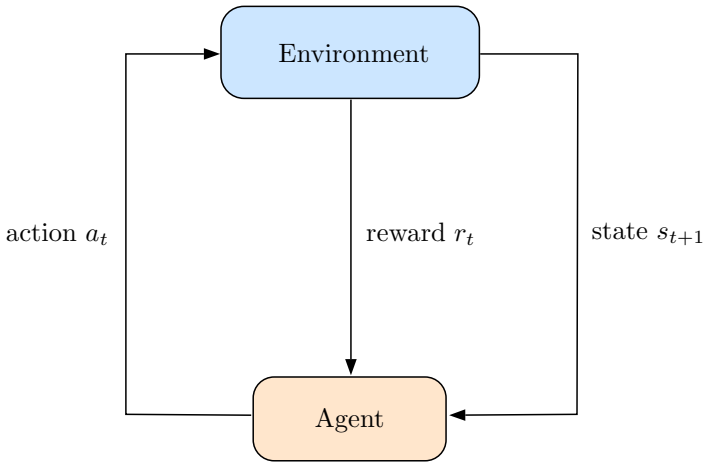

Figure 2.9: General RL setting.

action possible. RL blends those two types of interactions, making the resulting learning pattern different from the ones seen so far: the intelligent agent starts with zero knowledge (*tabula rasa*) about the context in which it is located. And, through an active exploration of the state space, looks for the optimal strategies. Starting from this, the most important challenge of any RL method is the trade-off between *exploitation* and *exploration*: the agent, in order to find the optimal strategy, needs to widely test all of the possible outcomes of its actions, so that the optimal ones can be finally chosen (exploration). On the other hand, to be able to recognize the best action, it has to collect enough information to maximize its target function (exploitation).

2.2.1 RL elements

Intuitively, the main components of RL are the *Agent*, that is the decision-maker, and the context/outside world called *Environment*. The two interacts through the exchange of actions, states and rewards (Figure 2.9). The environment can be affected not only by the agent's actions but also by external factors, that contribute to the complexity of the problem of interest. A cornerstone of RL relies on the formalization of the problem. In order to provide a discrete description of a stochastic environment for a sequential-decision making task, the usage of *Markov Decision Processes* (MDPs) becomes essential.

Definition 2.2.1 (Markov Decision Process)**.** We define as a Markov Decision Process the 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$ where:

- $\mathcal{S}$: is the set of all possible states of the environment;
- $\mathcal{A}$: is the set of all possible actions;
- $\mathcal{P}(s, a)$: is the probability that an action a will lead the state s to another state s';
- $\mathcal{R}$: is the immediate reward after the transition from state s to state s' conditioned to the action a being taken.

The function $\mathcal{P}$ is a probability distribution; hence, it satisfies the following condition:

$$\sum_{s',r} \mathcal{P}(s', r|s, a) = 1, \ \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{2.15}$$

The objective is to develop a mapping, called *policy* π to which the agent will stick so that a certain target function (that depends on the reward) is optimized.

Formally, we refer to a policy in the following way:

$$\pi : \mathcal{S} \times \mathcal{A} \longrightarrow [0, 1]$$

$$\pi(s, a) = \mathbb{P}[A = a | s = s]$$

In few words, at each step the agent observes the current state s and chooses an action a by sticking to its current policy π (with a certain amount of randomness added, that will be described later). As a result of the state-action pair (s, a), the environment returns a new state s' and a reward r, like displayed in Figure 2.9, that the agent will collect to optimize a target function.

Policy, Reward, Value Function

MDPs are divided into several sub-classes depending on the amount of information that the agent is able to observe. In our case, as we will explain later, the agent cannot access the majority of the model's features such as the transition probability or the existence of other agents in the environment. To deal with that, the agent needs to infer the environment's statistics through the observation of the state and the collection of the rewards.
The ***State*** $s \in \mathcal{S}$ describes the environment at the current time step. It can be partially or totally observed by the agent. Its formulation depends on the objective of the algorithm, it is then up to the developer to find the proper way of handling information within the state.
The ***Reward*** $r = \mathcal{R}(s, a)$ is a scalar value returned by the environment along with the new configuration (i.e. next state s'), after every action the agent has taken. The reward represents how the chosen action has affected the system. It is related to the immediate return. The learner does not know the specifics of the reward function's formulation $\mathcal{R}(s, a)$. However, it is capable to reinforce the goal if the mapping $\mathcal{R}(\cdot, \cdot)$ reflects what is the objective and/or how to compute it.
While the *reward* represents the immediate response of the system to the agent's action, the ***Value function*** $v(s)$ stores the long term expectation when starting in a state s and following π. The long term return is a discounted sum of rewards, starting from an instant t up to infinity. More formally, the return G starting from time t is defined as follows:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+1+k}, \tag{2.16}$$

where $0 \leq \gamma \leq 1$ is the discount factor. When $\gamma = 0$ we say the agent is myopic as it will take into account only immediate rewards; for a $\gamma = 1$ the agent equally weights every collected reward. We differentiate the value function into *state-value function* $v(s)$ and *action-value function* $q(s, a)$ depending on the variables affecting the agent.

Definition 2.2.2 (Value Function)**.** We refer to as State-Value Function $v_\pi(s)$ for the policy π to the expected long term reward of the agent by choosing the policy π starting from the current state s. Formally, for each time step $t \in \mathbb{N}$:

$$v_\pi(s) := \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+1+k}|S_t = s\right], \text{ for all } s \in \mathcal{S}, \tag{2.17}$$

where G_t is the expected discounted reward, and $\gamma < 1$ is the discount factor.
Similarly, we refer to as the Action-Value Function $Q_\pi(s, a)$ for the policy π the following quantity:

$$\begin{aligned} q_\pi(s, a) :&= \mathbb{E}_\pi[G_t|S_t = s, A_t = a] \\ &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+1+k}|S_t = s, A_t = a\right], \\ &\text{for all } s \in \mathcal{S}, a \in \mathcal{A}, \end{aligned} \tag{2.18}$$

where G_t is defined as before.

Note that both functions are defined for a fixed policy π. The main feature of such equations is that they satisfy the recursive relationship, something that became essential in solving MDPs through Reinforcement Learning. Starting from equation (2.17), we can write the return G_t recursively as:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+1+k} = R_{t+1} + \gamma G_{t+1}. \tag{2.19}$$

Plugging 2.19 into the Value-State function (2.17), we obtain:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t|S_t = s] \\ &\overset{(2.19)}{=} \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} \mathcal{P}(s', r|s, a)\left[r + \gamma v_\pi(s')|S_t = s\right] \end{aligned} \tag{2.20}$$

The equation displayed above is defined as the *Bellman equation for* v_π. Similarly, we define the Bellman equation for the action-state function q_π. Such equations show the relationship between the value of the current state $v_\pi(s)$ and its successor $v_\pi(s')$. In other words, we can look ahead at the next states and average all of the possible reward-state combination. In most of the cases, information like the transition probability of the system is not communicated to the agent. Nonetheless, such equations will be the fundamentals for the updating rule of the majority of RL-based methods.

Optimal Control

Solving a MDP means to find a policy such that the long term reward is maximized. We refer to such policy as the optimal $\pi_\star$. Another strength of the Bellman equation (2.20) is that they give a partial ordering to policies. That means that a policy π can be compared to other policies π' via their value function. Even if there might be more than one optimal policy $\pi_\star$, they all share the same state-value function $v_{\pi_\star}$, denoted as:

$$v_\star(s) = \max_\pi v_\pi(s),\ \forall s \in \mathcal{S}. \tag{2.21}$$

On the other hand, if we rewrite the state value function in terms of the action-value

$$v_\pi(s) = \max_{a \in \mathcal{A}} q_\pi(s, a),\ \forall s \in \mathcal{S},$$

and substitute it into the previous equation, we are able to re-formulate the Bellman equation without referring to a specific policy (in this case, the optimal/s). Formally:

$$\begin{aligned} v_{\pi_\star}(s) &= \max_a q_{\pi_\star}(s, a) \\ &= \max_a \mathbb{E}_{\pi_\star}[R_{t+1} + \gamma G_{t+1} | s, a] \\ &= \max_a \mathbb{E}[R_{t+1} + \gamma v_\star(S_{t+1}) | s, a] \\ &= \max_a \sum_{s', r} \mathcal{P}(s', r | s, a) \Big[r + \gamma v_\star(s') \Big]. \end{aligned} \tag{2.22}$$

The last equation takes the name of the *Bellman Optimality Equation* for $v_\star$. The result shows us how is it possible to retrieve an optimal policy without directly looking for it. In fact, having the optimal value function is enough for the agent to act optimally: it will need to just look at

the next state s' and choose the action that maximises the Bellman optimality equation.
Similarly, the optimal relation is provided for the action-value function, which makes the agent's search even better. At the cost of taking into account all of the possible actions for each state (and so by increasing the dimension of the variable state), the agent can simply choose the action that maximises the q-value function at the next state s':

$$q_\star(s,a) = \sum_{s',r} \mathcal{P}(s',r|s,a)\left[r + \gamma \max_{a'} q_\star(s',a')\right],\ \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{2.23}$$

However, it is quite difficult to solve the Bellman Optimal Equations. The reason behind is quite straightforward: a system of $|\mathcal{S}| \times |\mathcal{A}|$ equations requires a lot of time and memory, and sometimes the solution simply cannot be retrieved by a computer.
Thus, RL distinguishes two main paradigms to *approximate* the solution (which could be the value function and/or the policy) based on the complexity of the problem: we will refer to *tabular methods* for those of which rely on reasonably small action-state space, *approximate methods* otherwise.
For this thesis, we focused on the application of a well-known tabular method that is the *Q-Learning* technique. We now describe the general characteristics of such learning algorithm, as it will be later adapted to the problem of our interest in the next chapter.

2.2.2 Tabular-based algorithms: a Q-Learning case

The *Q-Learning* algorithm had its first development around the 90's, to solve optimal control problems. Like the name suggest, the algorithm approximates the optimal solution via the $q_\star$ function, based on a set of characteristics that define the method as one of the most versatile and sophisticated for tabular cases.
Q-Learning is a kind of model-free, off-policy, TD-learning method. We shall now explain the mentioned characteristics:

- *Model-Free:* by this term we intend all the techniques that do not include the knowledge of the system dynamics $\mathcal{P}$. Thus, the learning procedure relies entirely on the agent's actions and the collectable interactions with the environment;

- *Off-Policy:* the learner does not follow a specific policy to optimize but instead it maximizes its q-value function. By doing so, we are still acting optimally thanks to the Bellman Optimality Equation (Equation 2.23). This kind of algorithms relax the compu-

tations by removing the policy-iteration and by focusing on the value function approximation;

- *TD-learning:* temporal difference (TD) learning is a class of methods that uses raw experience to learn the objective. In contrast to Monte Carlo methods, the TD does not wait the end of the episode (an episode is the whole simulation in our case) to update the function but it bootstraps the outcome instead.

The reasons behind the choice of such algorithm are quite simple: the objective requires the implementation of multiple agents taking actions withing a shared environment without knowing the channel characteristics (*model-free*), as well as the presence of any policy at priori (*off-policy*), but only through the exchange of real-time data (*TD-learning*).
Like mentioned, by acting off-policy the agent must select an action according to its q-value. Generally speaking, such procedure is enriched with an exploration rate $\epsilon > 0$ that usually reassures the agent to visit all of the possible states and actions combinations. In practice, with probability ϵ a random action will be chosen, whereas the action that maximizes the Q-matrix at the current state will be chosen with probability $1 - \epsilon$. Such method is called ϵ-greedy. Afterwards, the agent collects the reward r and updates its q-matrix at the current state, action cell (s_t, a_t) by following the updating rule showed below:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r + \gamma \max_{a'} Q(s_{t+1}, a) - Q(s_t, a_t) \right], \tag{2.24}$$

where $\alpha < 1$ is the learning rate, $\max_{a'} Q(s_{t+1}, a)$ is the "look-ahead" phase and $\gamma < 1$ the discount factor of the latter. Generally speaking, the learning rate is defined at each time frame α_t, but in our case it is fixed for every instant i.e. $\alpha_t = \alpha$ for all t. A single-agent pseudo code of the Q-learning is given in Algorithm 2.1. The following theorem exhibit [15] the convergence property of the Q-learning method.

Theorem 2.2.1. Given a finite MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{A}$ and $\mathcal{S}$ are finite spaces, then the Q-learning algorithm converges with probability 1 if:

$$\sum_{t=0}^{\infty} \alpha_t < \infty \bigwedge \text{ all state-action pairs be visited infinitely often,} \tag{2.25}$$

where α is the learning rate.

Algorithm 2.1 Single-Agent Q-learning pseudo code.

Initialize the Q values and the hyper-parameters:

$Q(s, a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$

$\alpha, \gamma \in [0, 1], \varepsilon > 0$

for each episode t

Given current state s_t, choose action a using ε-greedy:

$x \leftarrow$ uniform random number in $[0, 1]$

if $x \leq \varepsilon$

$a_t \leftarrow$ random action from the action space $\mathcal{A}$

else

$a_t \leftarrow \arg\max_{a' \in \mathcal{A}} Q(s_t, a')$

end if

Take action a_t, observe next state s_{t+1}, retrieve reward r

Update Q value:

$$Q(s_t, a_t) \leftarrow Q(s_t, a) + \alpha \left[r + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a') - Q(s_t, a_t) \right]$$

Until convergence

end for

Another crucial remark that needs to be acknowledged about the Q-learning is its deterministic nature. Indeed, looking at Algorithm 2.1 we can observe that the agent will always choose, without considering the exploration phases, the action that maximises the current row of the Q-matrix. When applying such technique to a non-deterministic problem like the one presented in the previous section, a few modifications might need to be carried out. In the next chapter we will present the adaptation of Q-learning to the problem of interest.

Now that we have set the technical background, we can dig a little deeper into the problem. As it was stated form the beginning, we aim to develop ad-hoc strategies for each user in the network; by that we intend to implement a distributed learning procedure where each agent is an user interacting with the same receiver, that is going to be part of the environment.

We shift our focus to the setting of Multi-Agent RL, where single-agent properties do not hold anymore due to the interactions between the agents and the environment. We present in the following section the main challenges that this sub-field entails.

2.2.3 Multi-Agent RL: an overview

Multi-Agent Reinforcement Learning (MARL) [16], assess the involvement of multiple autonomous agents withing a shared environment. It is obviously an extension of the standard single-agent RL theory, but with many aspects and features that need to be accounted when dealing with it.
Multiple agents denote multiple learners that simultaneously update their target function, in a distributed manner. As a clear consequence, the environment's state is the outcome of the *joint actions* of all the interacting agents. MARL algorithms distinguished themselves for different levels of awareness of the agents in terms of each other. In the setting of our thesis, as will be explored in the next chapter, the agents do not communicate with each others, as the main receiver is the only entity acknowledged by them. Thus, each device is an independent learner that acts on behalf of its own objective, regardless of the others. Such setting takes the name of *Fully Decentralized Fashion*. We will consider the system, from now on, to be under such assumption.
Starting from this, we shall categorize two different kinds of information that each decision-maker may infer or collect:

- *Local Info:* with local features we intend all the statistics related to the single agent. In our setting: the throughput, the packet loss rate and the mean age of the single user. Such quantities are not shared in the fully decentralized setting as there is no communication between the agents;

- *Global Info:* with global features we intend all of the collective insights such as the mean age of the channel, traffic load or the system throughput. In a fully decentralized setting, such quantities cannot be transferred to the agents; thus, the learners may decide to infer or to approximate them with their own local info.

It is crucial to distinguish between the two in order to develop a specific goal-oriented intelligent system. Indeed, it is important to clarify that since a MARL framework is the aggregation of different agents with specific targets, the formulation of the goal becomes one of the main concerns.
MARL algorithms are organized depending on the interests of the agents: literature recognises fully cooperative problems, fully competitive or mixed tasks (nor cooperative or competitive). As it will be explained in the next chapter, this thesis focuses on the implementation of mixed tasks MARL.

Shifting from one agent to multiple ones, significantly changes the system formulation. Indeed, MARL problems cannot be expressed anymore in terms of Markov Decision Processes. To understand that, we have to recall that a MDP claims the Markov property i.e. the outcome of the environment state is the product of the action chosen in the previous state. Such property is no longer valid in the multi-agent case: the environment's state is the consequence of the joint actions of the agents and in some cases it may depend not only on the immediate previous choices but also on past interactions.
To properly describe a MARL scenario, *Stochastic Games* (SGs) come at help from the Game theory field, [17]. We may now give a formal definition of a stochastic game.

Definition 2.2.3 (Stochastic Game)**.** A *Stochastic Game* [17], named also *Markov Game*, is a tuple $(n, \mathcal{S}, \mathcal{A}_{1,\dots,n}, \mathcal{P}, \mathcal{R}_{1,\dots,n})$, where:

- n is the number of agents within the environment;
- $\mathcal{S}$ is the set of all possible environment states;
- $\mathcal{P}$ is the function describing the dynamics of the system;
- $\mathcal{A}_i$ is the set of all actions available for the i-th agent;
- $\mathcal{R}_i$ is the reward function of the i-th agent.

It is important to highlight that the agent reward functions are decentralized as well even if the punishment or award are going to be based on the agent's action *and* its interaction with the other users. The class of MARL algorithms based on individual rewards takes the name of *Reward Sum* [18]; on the other hand, the class of methods that provides to each agent the same reward function takes the name of *Shared reward*, but the latter won't be the case of our interest.

Reward Design

Generally speaking, it is well-known how the reward formulation has a crucial influence in any RL method. In a multi-agent setting, the reward can play an additional role that may regulate the social behavioral pattern of the system. In simpler terms, the agent may be able to act on behalf of the others if the reward is oriented on a fundamental question that is *what the goal is*. Here the notion of global and local info is again crucial in order to distinguish *selfish* or *lazy* strategies. In the work presented by [19], the authors emphasize how the reward formulation

can assign the sense of community and individuality to the agent's approach.
In a global reward setting, the agents receive a reward related to the global outcome of the system, which can be, in our problem, the mean age of the channel or the current load. By doing that, the agent is not able to recognize its own contribution to the system: a positive o negative reward might be related to the action of another agent. In this way *lazyness* is encouraged. An agent may get positive/negative rewards thanks to another user, and so the obtained strategy might be satisfiable enough. Such behaviors rarely lead to global solutions, but they find instead sub-optimal strategies or even worse, they could lead to an unfair share of the environment.
On the other hand, in some cases, returning only local rewards might not be efficient as well. If the agent does not acknowledge any insights about the others, *selfishness* is certainly encouraged. In this way, the agent will not help each other but act greedily instead.
Such literature review has inspired the making of the algorithm's reward which is going to be explained in the next chapter. We now continue to list the main challenges of a MARL setting.

Joint Learning Goal, Non-Stationarity

In a reward sum setting, each agent has an individual reward mapping as well as a target function to optimize. In the single-agent case, the goal is to maximize the long term reward. In the MARL case, the objective can be often hard to formalize. Generally speaking, the goal of a MARL problem is to find the Nash Equilibrium (NE) of the correspondent Stochastic Game. A Nash Equilibrium is a joint-strategy $(\pi_1^\star, \dots, \pi_n^\star)$ of the game such that nobody is encouraged to deviate from it. In terms of SGs, in order to find an equilibrium, a proper definition of the game has to be presented. This task may be vague sometimes [20]. Moreover, it has been shown how finding a NE in a general-sum stochastic game is actually hard to do in practice. This adds to the convergence analysis concerns about whether the obtained policy is a NE or not [18]. We know from theory that, if we keep the agent strategies fixed except for the i-th user, then the i-th agent will converge to the optimal solution under the Q-learning algorithm. However, if the learning is simultaneous, the environment becomes non-stationary, and there is no guarantee that the agents will jointly converge to a solution. Sub-optimal solutions, or weaker equilibrium, can be found instead.

MARL agenda

We have seen a broad overview of the challenges that a multi-agent setting can bring. What arises from it is that sometimes the formulation of the problem itself can be hard and so the agent's objectives may result vague.
In the survey [21], the authors provide a well-formed agenda for MARL problems. In order to properly develop a multi-agent framework, a series of *questions* needs to be made up. Among the ones presented in the study, the *AI agenda* is considered the most interesting and representative in terms of categorizing the agents. It asks which strategy should be the best one for an agent when the others are fixed. Such query is intended to model the agents in the best way possible, and that can be expressed in many framework features: from the modulation of the state and actions space to the reward function. Indeed, speaking about the reward mapping, the authors of [19] explain how in multi-agent cases the reward function can be tricky to model. In single-agent RL the reward has always been shaped around *how to achieve the goal*: e.g. often using a negative reward of -1 when the agent is not acting for its good, or 1 otherwise. In MARL setting instead, the authors focus on the reward designation of *what is the goal*. As mentioned before, the latter can have an impact on the policy in terms of global and local interests.

2.3 Related Works

This chapter concludes with an overview of the related works present in literature. Given the interdisciplinary nature of the topic, is it possible to approach the researchers works through layers.
Starting from random access channels, a study on the performance of slotted ALOHA-based contention was first proposed in [22], [23]. These contributions recently triggered a flourishing line of research, exploring the implications of different aspects on AoI. Like mentioned in the previous dedicated section, AoI-based policies gained attention as a promising technique to improve SA-like protocols [12].
Different kind of approaches have been adopted, like in [24], where the authors tackle the AoI optimization problem from a game theoretic point of view. In [25], the authors analyze a random access protocol and test fixed and dynamic transmission policies for the avoidance of collision deadlocks. Distributed settings have been tested as well, as in [26], where game theoretic solutions are provided for different IoT applications as vehicular networks and other real time scenarios. Decentralized policies have been analyzed in [27], where two strategic players

take into account a cost each time an update is sent. Similarly to the work that will be presented in the next Chapter, the nodes independently retrieve their policies, but via solving the NE of the game rather then through a learning phase. Interesting insights from an adversarial analysis are presented in [28], where two nodes adopts defence policies in a vehicular network scenario which could find application in the AoI area.
Research has later been extended with the rise of AI, specifically with Reinforcement Learning, on the optimization of random access protocols. In the work of [29] and [30], the authors apply Q-learning on framed-SA for a throughput-based optimization; those studies show the weaknesses and strengths of Q-learning convergence in stochastic environments. In contrast, in our work the time horizon has been divided solely by slots rather than frames made up of slots, and the reward has been based on the mean AoI rather than the standard $+1/-1$ reward/punishment system. In [31], on-policy algorithms like SARSA and Upper Confidence RL (UCRL2) are compared together with Deep RL as well, on the quality of transmission scheduling for AoI-aware devices in a HARQ protocol. SARSA and UCRL2 are on-policy algorithms [15], while Q-learning is an off-policy algorithm. The work has been taken as a benchmark when looking for the proper RL-based method for the handling of our problem.

Deep Reinforcement Learning has recently gained attention thanks to its State-of-the-Art results in several automation fields. Given its remarkable improvements with respect to standard RL algorithms, SA-based protocols have been tested to overcome benchmark results. In [32], the authors implement a Deep Reinforcement Learning method to jointly optimize the wireless energy transfer (WET) and minimize the Age of Information of the overall system. Similarly to our work, a Q-learning algorithm is used to reinforce the channel access strategy. In contrast to this thesis, the authors utilize a Neural Network (NN) to firstly extract the useful features from the environment. In [33], a DRL paradigm is implemented to improve the mean AoI of the system. The authors introduce the role of urgency of an user as well as the utilization of a buffer, from which mean values of age are computed with the assumption of having Poisson arrival rates. Results show interesting policies based on their current age and their current urgency of transmission. However, the number of devices is known to each agent as well as the Poisson arrival rate of packets.

3
AoI-Q-ALOHA

In this chapter, we formalize the framework of this thesis along with its objectives and the algorithm chosen to tackle the problem. By incorporating the background of the previous Chapter, we give a formal definition of the topic.

This thesis focuses on the optimization of a slotted ALOHA-like protocol in terms of the overall freshness of data flow - Age of Information - under the paradigm of Q-Learning. In a fully decentralized fashion, the users develop ad-hoc strategies for the minimisation of their local mean AoI without any communication with the others, expect for the central receiver. Team behavior is encouraged through a tailored individual reward designation. Throughput, Fairness and average Age of Information values are compared to standard SA performances as well as to the threshold-ALOHA policy.

3.0.1 Why Q-learning

Before digging into the system model, it is important to articulate the reasons behind the choice of a tabular-based algorithm. With the rise of Deep Reinforcement Learning (DRL) [34], the first idea to optimize a complex real-time task like the one at hand would be to use one of the state of the art methods like DQN [35], where Q-learning is coupled with a Deep Neural Network (DNN) to handle high-dimensional spaces. However, training a DNN comes with a certain cost as well as the energy consumption that could require efficient memory and stor-

age capacity. The focus of this thesis is to optimize an already built-up environment that is a *wireless monitoring network*, where each user is a sensor tracking a specific process. Monitoring shall last as long as possible as each device is located in different spatial locations; thus, eventual maintenance should be reduced at its lowest possible rate. To do so, tracking systems are often based on battery powered, low-complexity devices.
Starting from this, the optimization of an already low-cost setting cannot be done with a significant shift in implementation prices, otherwise the objective would not be practical-oriented. Moreover, Q-learning still represents a method of choice for many relatively small-dimensional spaces.

3.1 System Model

Consider a network of n devices, each of them monitoring a specific process, and sharing the same receiver. We consider a slotted-ALOHA protocol to manage the link layer communication system. Each node sends updates through the transmission of data packets to the main receiver, which is shared by the users. Time horizon is divided into slots, and each user is slot-synchronized. We consider a channel collision model [8], where if two or more users transmit a packet in the same slot, then a collision occurs, and the data is considered lost. On the other hand, if a packet is the only one sent in given slot, then the transmission is said to be successful. We consider the "generate-at-will" model [12], where each node has a fresh data packet at every slot. The transmission of packets is attempted by following a probability τ. We consider the throughput as the basic metric to evaluate the rate of successfully sent packets over an horizon time. However, the throughput lacks on utility when it comes to track the freshness of data, which becomes the main concern when talking about monitoring systems. To track the receiver's freshness knowledge about a specific process, the Age of Information (AoI) comes at help. For each slot t, the AoI of an user is defined as:

$$\forall i,\ \delta_i(t) := t - \sigma_i(t), \tag{3.1}$$

where $\sigma_i(t)$ is the latest slot in which the receiver has been successfully informed about the i-th process. The AoI is linked to the throughput as the reset of the age comes with a successful deliver. Each user is able to track its own AoI over time thank to the acknowledgment feedback (ACK) sent by the sink to every user at the end of each slot, notifying about the outcome of the latest slot contention. Current AoI, mean AoI and single node throughput are tracked by

each user throughout the simulation.
Each user is an intelligent system (*agent*) consciously interacting with the sink, and unconsciously interacting with the other agents. For each user, the central receiver and rest of the network are part of the *environment*. As mentioned, in our multi-agent setting (MARL) it is not true that the totality of the environment is observable for the agent, and so the framework presented falls under the name of *Partially Observable Markov Decision Processes* (POMDPs). Specifically, each agent can only observe its own age $\delta_i(t)$ and the outcome of the latest channel contention. More formally, we define the state of each agent in the following way:

$$\forall\, i, \delta_i(t) = \begin{cases} \delta_i(t) & \text{if } \delta_i(t) \leq \Theta \\ \Theta & \text{otherwise} \end{cases} \tag{3.2}$$

the truncation at Θ is for computational reasons, and is set high enough for the agent to explore it only sporadically once convergence has been reached. To be consistent with notation, we mention that the if the overall system is considered as a whole, then the state of the environment at each slot t is the following:

$$s(t) = (\delta_1(t), \ldots, \delta_n(t)), \tag{3.3}$$

but as previously mention, each device can only observe a portion of it i.e. its own AoI.
As far as the possible actions, each agent has the same set of actions for each slot:

$$a_i(t) \in \mathcal{A} = \{wait,\ trans\}, \tag{3.4}$$

where $wait$ means for the node to remain silent, access the channel by using SA otherwise. The *joint* actions of all agents $(a_1(t), \ldots, a_n(t))$ will interact with the environment, which will return to each agent an individual reward $r_{i,t}$ based on their contribution in the latest channel contention, and their individual objective. We recall that the presented system model acts in a fully decentralized manner (Fig. 3.1) so each agent is totally detached with the others, with no communications expect for the central receiver.

Objective

We aim to develop a strategy which leads to good performances in terms of mean channel AoI as well as the individual AoI of the nodes. However, it is important to first analyze the conflicts

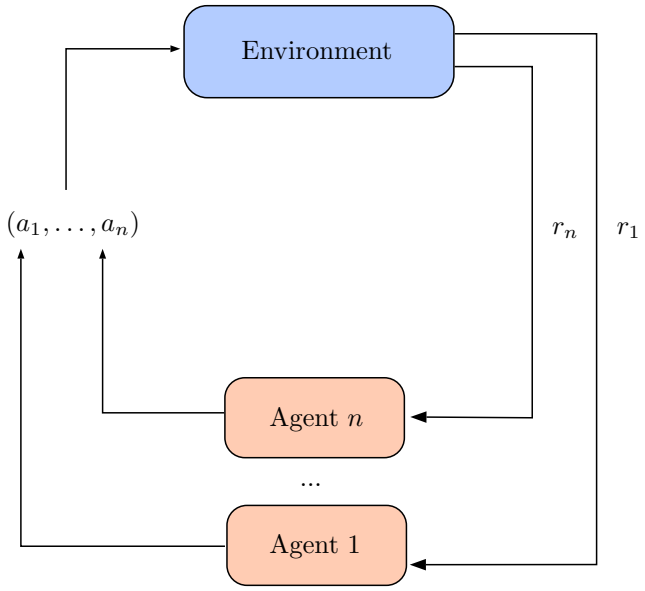

Figure 3.1: MARL setting in a POMDP.

between the agents interests.

Generally, a MARL problem is categorized based on objectives of the agents. In a cooperative setting, each agent has the same goal, that, in our case, would be about the mean channel AoI. Here, each agent would receive the reward from the *same* mapping $\mathcal{R}_i = \mathcal{R}\ \forall i$, and everyone would have the same objective.

In a competitive setting instead, each agent would have an individual target and in order to achieve it they would have to act against the others.

Furthermore, when the setting is a mix of the two, the problem takes the name of *non-cooperative and non-competitive* games, or *general-sum* games. The latter takes the name from Game Theory and is related to the arbitrary nature of the payoff functions. The framework presented falls under the last category for the following reasons:

competitiveness : it is obvious that each agent would prefer to reduce its mean age. In order to do that, the user may choose to transmit whenever it has the chance, regardless of the others. However, acting selfish in a multi-user network indirectly requires the cooperation with the others to avoid collisions. That is why, the ACK signal intrinsically pushes the nodes to act for the good of the channel, in order to reduce their age. Thus, it is not proper to define the problem as a competitive one;

cooperativeness : a complete game would require the total knowledge of the channel's topology, as well as the current age of each node, so that everyone can cooperate on when and how to transmit. However, we know that such level of cooperation is not achievable when only a small portion of the system is observable by each agent. Moreover, the long term objective of each agent cannot include the channel AoI as it would require the knowledge of it (decentralized setting). One of the main objective of this thesis is to let the single agent learn by itself only with the information that it could properly retrieve. In conclusion, the problem cannot even be defined as cooperative.

Due to the lack of communication among the agents, we define the game as a general-sum game, or as a non-cooperative and non-competitive game. We will show in the next sections how community is encouraged with the use of local information and the ACK signal. Results and discussions will follow.

3.2 The Algorithm

For the development of distributed policies, *Q-learning* has been chosen thanks to its simplicity in terms of computational costs, memory and its fast convergence properties (it handles very quickly small-dimensional spaces and it is much more simpler than policy-based algorithms). Q-learning is a value-based algorithm as the agent learns its policy through the optimization of the Q-matrix. Thus, each agent will develop its access channel strategy through the exploration and exploitation of the $Q_i(\cdot,\cdot)$ matrix. At each slot, the agent will access its Q_i matrix at the current state/*row* $Q_i(\delta_i(t),\cdot)$ and subsequently choose the action/*column* that maximizes the $Q_i(\delta,a)$ cell. The memory requirements will be the following:

$$\text{for every agent } i,\ |Q_i| = \Theta \times 2; \tag{3.5}$$

where Θ is the state-space dimension and 2 is the action-space dimension. Recalling that the state is presented by the current age δ, the state-space defines an age-limit i.e. Θ which is large enough for the agent to sporadically reach it once convergence has been achieved. Like stated in literature, if the state-action space is reasonably small then the Q-learning is the best fit for the solution of the algorithm.
We shall recall that the Q-learning is a deterministic algorithm i.e. it chooses the action that maximises the matrix state-row at each instant. Because our goal is to define *probabilistic* poli-

cies on how to access the channel, we modify the standard procedure by adding a contention probability each time the action $trans$ is chosen. More specifically, the node will attempt a transmission with probability τ when the action chosen is $trans$, and stay idle otherwise. Any further details about its updating rule are explained in the next section.

3.2.1 Reward Formulation

Since the goal of this thesis is to develop access strategies solely based on local features, the function $\mathcal{R}$ has been shaped in a way that the agent is guided to minimize a localized target. Inspired by the work of [19], the reward has been tested on different levels of credit assignments, i.e. on different levels of global and local contributions. An imbalance of the two spheres can lead to laziness, if the node gets relatively good rewards without recognizing its true role, or selfishness, if the node gets only individual reward without crashing with other interests [15]. In our case, the only *global* trend of the channel, that can be observed by each agent, is represented by the ACK signal, so it will be integrated in the reward function. Thanks to that, the agent is able to track all of its *local* features such as its throughput rate, collisions rate and mean AoI over time. Let us now focus on the actions available: $\{wait,\ trans\}$. It is important to remark that the $wait$ function behaves differently from the other: the first one is deterministic, while the other triggers an additional level of randomness resorting to the transmission probability. Thus, additional conditions must be defined in the second case.
We shall recall the reward is a function of the current age/*state* and the action and so the notation will be, from no on, as $\mathcal{R}(\delta(t), a)$. If the node chooses the action $wait$ then the reward will be based on its local mean age:

$$\mathcal{R}(\delta(t), wait) = 1 - \frac{\delta(t)}{\Delta(t)}, \tag{3.6}$$

where $\Delta(t) = \frac{1}{t}\sum_{k=0}^{t-1}\delta(k)$ is the mean age of the agent up until time t, an $\delta(t)$ is the current age at time t. The reward is positive when the age is below its mean value, negative otherwise. In this way the agent is encouraged to stay silent while its age is relatively low and then gradually encouraged to transmit as its age gets higher and higher than its mean age. The relation is inspired by the threshold-ALOHA policy [12] that sets the nodes silent when their age is below a fixed threshold. Here, because the agent has no prior knowledge, it is able to use only its local insights like its mean age. Such procedure will lead to the emergence of a threshold Γ_i for the i-th node, possibly varying with time and derived in a fully decentralized way.

If the node chooses the action $trans$ instead, the outcome is not deterministic anymore but depends on the outcome of the probabilistic contention. More specifically, we set the reward as:

$$\mathcal{R}(\delta(t), trans) = \begin{cases} \dfrac{\delta(t)}{\Delta(t)} - 1 & \text{if success} \\ -1 & \text{if collision} \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

where, in case of success, the reward is negative if the age is below its mean AoI, linearly increasing otherwise. The reasoning is similar to the one presented for the $wait$ action: we would like the node to start transmitting when its age its above a certain threshold so that the older users will have the chance to reset their age. Again, because the threshold has to be learnt from experience and from local computations, the mean AoI is set as reference point. However, assigning the individual $\Delta(t)$ it is not totally local as its value depends on the choices of the single agent but also on the interaction with the other participants in the channel. Lastly, the reward is set to 0 when, after choosing to transmit, the node actually remains silent. Because such outcome relies entirely on a varying transmission probability, the assignment to a non-zero value could bring the node's policy to diverge from its objective. In conclusion, such reward system should indirectly encourage the agent to act for its own AoI minimization and for the good of the community.

3.2.2 Adaptive Transmission Probability

In this section we present the modification to the standard Q-learning procedure when choosing an action.

The $trans$ action cannot be treated deterministically because of how the algorithm is defined. In fact, if the agent were to transmit with probability 1 whenever indicated by its Q-matrix, then the rate of successes would be too low in comparison with the rate of collisions. More specifically, the agent would collect a significant amount of negative rewards, which will quickly affect its learning process and will let it to remain silent the majority of the time, and we know how letting the channel to stay idle is not desired. Moreover, acting deterministically when transmitting does not match to the reality of the problem (random access channels). To do so, an *adaptive transmission probability* is computed alongside the Q-learning. Each time the agent chooses to transmit, the channel contention is attempted with a probability τ_t which is adapted over time. Specifically, each agent will tune its access probability based on its successes

and collisions as follows. Starting from a random initialisation in a bounded range, the probability is updated each time the node transmits a packet, and depending on the outcome, the channel contention probability is increase or decrease by a fixed value w:

$$\text{for each transmission at time t: } \tau_{t+1} = \begin{cases} \tau_t - w & \text{if coll} \\ \tau_t + w & \text{if succ} \end{cases} \tag{3.8}$$

Similarly to the work of [36], where the policy is updated alongside with the Q-learning, the probability τ increases when the agent is in a "winning state" ($+w$), decreased when is "losing" ($-w$). The computation of τ is obviously affected by the trend of the Q-learning, but not vice-versa: under the proper set of parameters w, the algorithm quickly finds the equilibrium and so the moving probability simply speeds up or slows down the convergence process.

3.2.3 Hyper-Parameters

Finally, we present the hyper-parameters in our Q-learning and identify their potential influence in the learning process:

Exploration Rate $\epsilon > 0$: like in every RL method, the exploration rate plays an important role in the convergence. As stated in the previous chapter, the Q-learning procedure converges with probability 1 if all states are going to be visited infinitely often. The term ϵ ensures that such condition is satisfied. In our case, we would like to visit frequently enough all of the cells of the Q-matrix. Luckily, the state-action space dimension is relatively low, and the slot contentions are quite repetitive. Thus, such property is expected to be satisfied. Usually set as 0.01, the influence of it will be discussed in the next chapter;

Discount Factor $0 \leq \gamma \leq 1$: integrated in the updating rule of Q-learning (Equation (2.24)), the discount factor balances the trade-off between short-term and long-term rewards. With a 0-approaching term, we prioritize short-term interactions, while for values close to 1, long-term dependencies are maximized. In a complex framework like ours, where the randomness of channel contention removes

any clear links between the agents interactions, the discount factor can have an impact on the convergence speed;

Learning Rate $0 \leq \alpha \leq 1$: related to the convergence speed, the learning rate defines how quickly the agent wants to incorporate new information. Low values of $\alpha \ll 1$ will slow down the algorithm as small changes will be carried out. High values of $\alpha \gg 0$ will speed up the converge as new values are widely integrated in the updates. A trade-off is surely needed. Discussions on its influence on convergence are carried out in the next chapter.

3.2.4 AoI-Q-ALOHA

The proposed algorithm takes the name of *AoI-Q-ALOHA*. The pseudo code is shown in the Algorithm 3.1. The name merges the Q-learning (*Q*) and the optimization of slotted-ALOHA protocols (*ALOHA*) in terms of Age of Information (*AoI*).

3.2.5 Centralized AoI-Q-ALOHA

In order to enrich our survey, a *centralized* version of the AoI-Q-ALOHA has been implemented, whose name takes - AoI-Q-ALOHA, bound. The latter differentiates itself from the original version for the amount of feedback returned by the receiver: we suppose each user being able to observe the current age of *every agent*. Starting from this, each user will use its own Q-matrix to estimate the number of active nodes when it chooses to transmit: for each observed age, if the agent's Q-matrix row would choose to transmit, then the considered node is counted as active. Then, the agent will access the channel with probability $\frac{1}{\#active\_nodes}$. Such version clearly shares a significant amount of knowledge, and so we expect from it an outperform in terms of mean AoI of the original AoI-Q-ALOHA. The centralized version has been tested so that a decentralized version could be designed by gradually removing shared information from it.

Algorithm 3.1 AoI-Q-ALOHA

Initialize the Q matrices and the hyper-parameters:

$Q_i(\delta, a) = 0$ for all agent i, $\delta \in \mathcal{S}$, $a \in \mathcal{A}$

$\alpha, \gamma \in [0, 1]$, $\varepsilon > 0$

for each slot t

for each agent i

Given current age $\delta_i(t)$, choose action $a_i(t)$ using ε-greedy:

$x \leftarrow$ uniform random number in $[0, 1]$

if $x \leq \varepsilon$

$a_i(t) \leftarrow$ random action from the action space $\mathcal{A}$

else

$a_i(t) \leftarrow \arg\max_{a' \in \mathcal{A}} Q_i(\delta_i(t), a')$

end if

Take action $a_i(t)$, observe next state $\delta_i(t+1)$, retrieve reward r_i, retrieve ACK signal

Update transmission probability τ:

$$\tau_{t+1}^i \leftarrow \begin{cases} \tau_t^i + w & \text{if node has transmitted and } ACK = 1 \\ \tau_t^i - w & \text{if node has transmitted and } ACK = -1 \\ \tau_t^i & \text{else} \end{cases}$$

Update Q value:

$m \leftarrow \gamma \max_{a' \in \mathcal{A}} Q_i(\delta_i(t+1), a')$

$Q_i(\delta_i(t), a_i(t)) \leftarrow Q_i(\delta_i(t), a_i(t)) + \alpha \Big[r + m - Q_i(\delta_i(t), a_i(t)) \Big]$

Until convergence

end for

end for

4

Results and Discussion

This chapter is dedicated to the discussion of the results obtained from network simulations of the AoI-Q-ALOHA algorithm, as detailed in the previous chapter. We will first start with an overview of the main features of the algorithm on a fixed channel configuration, to then continue with an exhaustive survey on the influence of the major hyper-parameters. Moreover, observations will be provided on the role played by different network sizes. A comparison with the standard slotted-ALOHA and threshold-ALOHA policy will be given in all settings.

Setup

The generic setup used for the majority of the simulations is described in the Table 4.1. Recalling the goal of this thesis being on massive wireless networks, the user population has been set low enough to carry the computations within a reasonable time, but also high enough to not fall into the case of deterministic time division, where each node periodically gets an exclusive share of the channel. Thus, the chosen benchmark size is 100 nodes.

As mentioned, all of the results will be compared with performance of slotted ALOHA as well as threshold-ALOHA [12]. For the latter, as explained in Chapter 2, the protocol parameters in terms of threshold and channel access probabilities are set to the optimal values retrieved from the dedicated simulations. We shall recall their benchmark values in Table 4.2.

Besides the minimization of the mean AoI, we consider other desirable properties that the system shall exhibit:

Parameter	Value
network size, n	100
learning rate, α	0.1
discount factor, γ	0.1
exploration rate, ϵ	0.01
maximum AoI, Θ	700
adaptive step for τ	$w = 0.005$
simulation time (slots)	10^7

Table 4.1: Setup of the main simulations

Fairness ≈ 1 : the fairness index shall be kept as close as possible to 1, so that competitiveness and selfishness among users do not arise. In a MARL setting, an improper reward function could easily lead to Nash Equilibria where one or few nodes get exclusive utilization of the channel, leaving no room for other agents. The latter usually happens when the gap between the punishment and the reward is too wide: a node could get stuck in a series of entirely negative rewards, from which an escape would be impossible to carry out. Thus, the Jain's fairness index, introduced in Chapter 2, has been used as a quality indicator for the algorithm tests;

Throughput : we now how maximizing the throughput does not necessarily improve the AoI. As a consequence, we focused our work on the minimization of the channel AoI. Throughout the simulation, the throughput is kept monitored to assess eventual losses.

	Slotted ALOHA	threshold-ALOHA
Normalized mean channel AoI, $\overline{\Delta}$	2.72	1.47
Traffic, G	1	0.90
Throughput, S	0.368	0.365
Fairness, $\mathcal{J}$	0.99	0.99

Table 4.2: Performances values obtained from dedicated simulations which will be used as benchmark throughout the discussion of results.

Normalized mean AoI

We begin with the discussion of our main concern: the mean channel's Age of Information. In the Figure 4.1, the evolution over time slots of the normalized mean AoI $\overline{\Delta}(n)$ is shown. The dashed lines represent, going from the highest to the lowest value, the slotted ALOHA and threshold-ALOHA best normalized mean AoI respectively. The pink curve represents the evolution over time of the mean AoI under the AoI-Q-ALOHA algorithm; it is clear how convergence is quickly reached just after few thousand slots. This behavior is expected from a Q-learning paradigm. For a network of 100 nodes, the mean AoI converges to a normalized value of $\approx 1.7\ [\frac{slots}{node}]$, widely improving the SA performance by the 40% and leaving a 13% gap from the centralized threshold-ALOHA policy. We shall remark the the latter takes advantages of the knowledge of the network topology, while our setting is implemented solely sharing the ACK signal from the receiver. Moreover, as mentioned in the previous Chapter, a centralized version of the algorithm - AoI-Q-ALOHA, bound - has been tested as well, to assess the influence of information sharing among the agents. As expected, by making each agent able to observe the age of the remaining agents gives the chance to the learner, to estimate the number of active nodes so the transmission probability will be tailored for the exact slot time. The centralized algorithm converges to a normalized value of $\approx 1.4\ [\frac{slots}{node}]$, outperforming not only

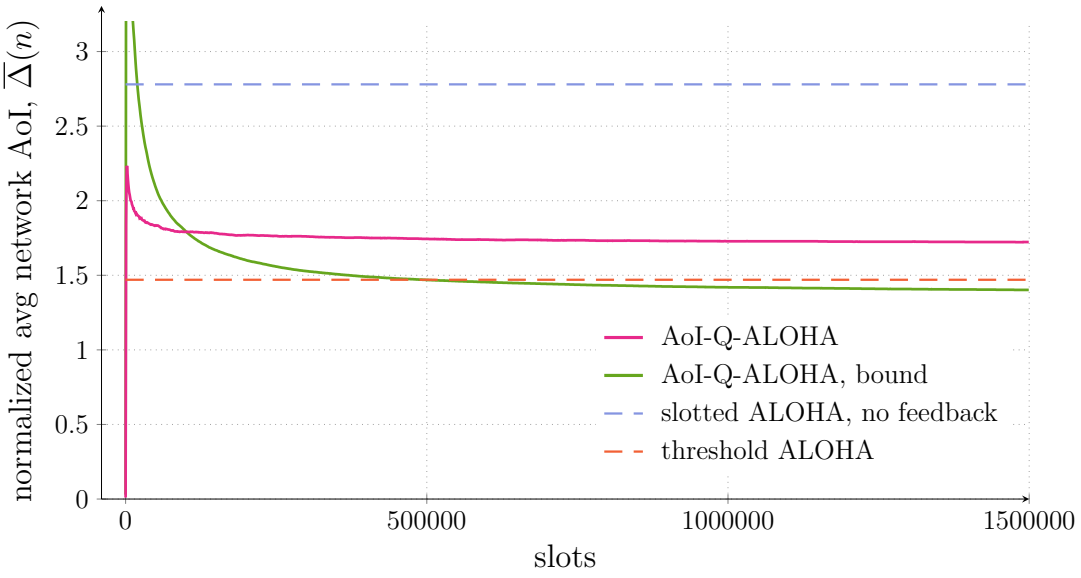

Figure 4.1: Evolution of mean AoI over time. Simulation held for a network of 100 nodes. Standard SA and threshold-ALOHA performances are displayed by the dashed horizontal lines.

	throughput, S	fairness, $\mathcal{J}$
AoI-Q-ALOHA	0.35	0.99
Threshold-ALOHA	0.36	0.99

Table 4.3: Throughput and fairness index at convergence of the AoI-Q-ALOHA algorithm and threshold-ALOHA policy on a network of 100 nodes.

the original AoI-Q-ALOHA algorithm but also the threshold-ALOHA policy, that we recall having a normalized mean AoI of $\approx 1.47\ [\frac{slots}{node}]$ for a network of 100 nodes.

Finally, Jain's fairness index and system throughput are displayed in the Table 4.3. As expected, a fair channel with a remarkable throughput system is obtained; just a small degradation on the latter, if compared to the reference results of SA and threshold-ALOHA shown in Table 4.2.

Structure of the Q-Matrix

The characterization of the Q-matrix is clearly what distinguishes the Q-learning from other RL-based algorithms; it displays how the reward has shaped the policy throughout the simulation. In Figure 4.2, the trained matrix of a sample node is shown. The x-axis represents the states i.e. the current age δ, while the y-axis represents the available actions, that in our case are the same for every state, i.e. $wait$ and $transmit$. The matrix is intentionally displayed as bi-color, to remark the deterministic nature of the algorithm: for each state/*row*, the agent chooses the action/*column* that maximizes the current $(state, \cdot)$ row. The max-action is represented by the white colour, black otherwise. As we exhaustively explained in the previous chapter, the reward mapping has been designed so that a threshold-policy would arise from it. For the presented setting of 100 nodes, Figure 4.2 shows how the agent chooses to wait until its age is around 200, and start transmitting above it. The obtained threshold is a sub-optimal solution if compared to the threshold-ALOHA policy, whose optimal parameter for a network of 100 nodes is around 220 slots. Nonetheless, we shall remark that in AoI-Q-ALOHA the node learns the policy only by collecting *local* rewards and by partially acknowledging the channel outcome slot by slot, without any assumption on the number of agents within the network. Looking at an unfiltered picture of the Q-matrix, i.e. without forcing the white/black color code, in Figure 4.3, the true values of the Q-matrix show a gradient of magnitude on the quality of action per each age value. First, let us focus on the *wait* column: it is smoother than the *trans* one and the reason simply lies on the consequence of the actions itself; each time a

node chooses to transmit instead of waiting, it is going to interfere with the other agents and so the outcome is probabilistic. Thus, the long term reward of the transmit action will contain negative and positive rewards, depending on the amount of collisions and successes collected during the simulation. On the other hand, the waiting action, by being a deterministic action, does not interfere with other rewards/punishments and so the long term reward is the value of the mapping:

$$\mathcal{R}(t) = 1 - \frac{\delta(t)}{\Delta(t)}.$$

Moreover, let us highlight how the matrix becomes noisy as the age increases. Such phenomenon is given by the sporadic visits on those states as convergence arises. Indeed, once the agent has reached its steady equilibrium, it is less likely that its age will reach values higher than double its mean age.

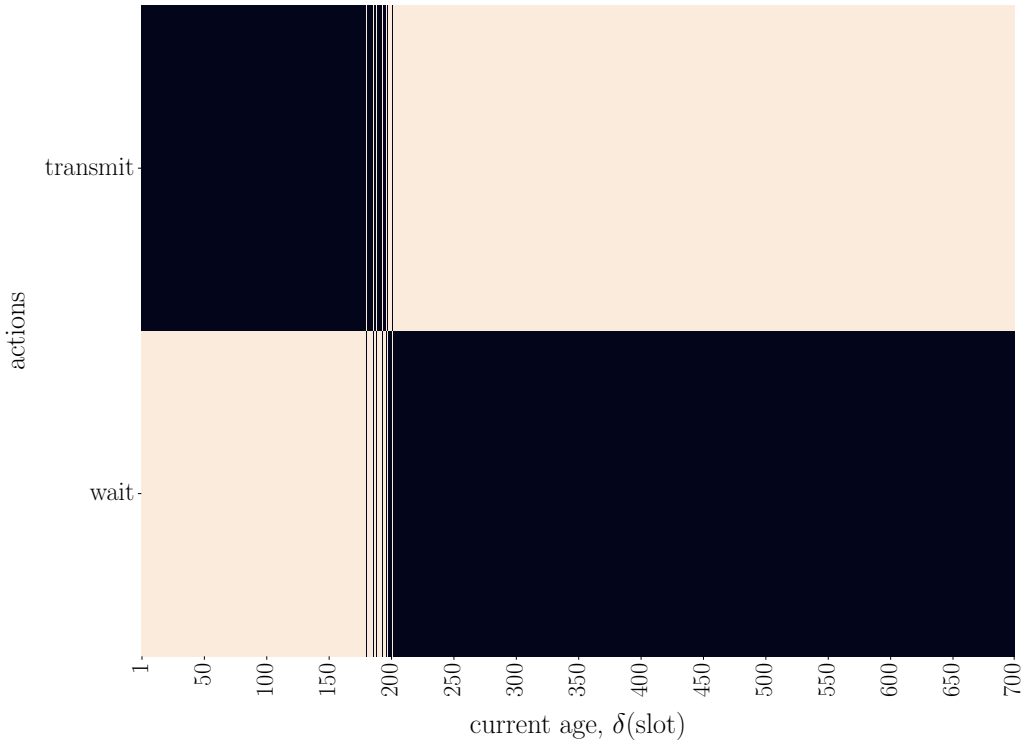

Figure 4.2: Single Q-Matrix of a sampled node trained on AoI-Q-ALOHA algorithm at convergence. Every white cell indicates the action that maximizes the specific row.

Let us now shift our focus to the *average matrix* $\overline{Q}$ of the network. In our objective, a co-operative goal is defined, that is the minimization of the channel AoI; however, each agent is aware of the minimization of solely its own AoI as the decentralized setting cannot reveal the presence of the others. Thus, the mean matrix describes resulting unbalances in the policies space. In the Figure 4.4, the overall Q-matrix is displayed as the two Q-values actions in terms of the age. On the x-axis, the age, δ is represented; along the y-axis, the $\overline{Q}$-values of the actions $wait$ and $trans$ are exposed. The two curves are computed via the following equation:

$$\forall a,\ \overline{Q}(\delta, a) = \frac{1}{n} \sum_{i=1}^{n} Q_i(\delta, a)\ \forall \delta$$

The threshold behavior clearly emerges at the intersection of the two curves. The light coloured vertical lines represent the lowest age at which each agent decides to access the channel. They

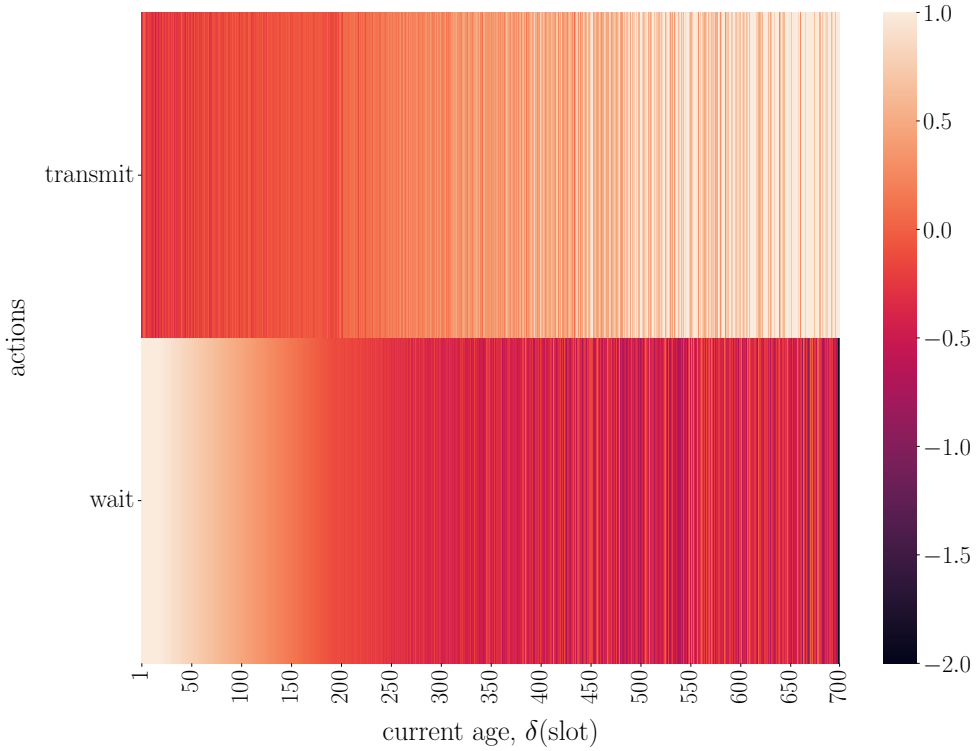

Figure 4.3: Raw Q-matrix values of a sampled node trained under the AoI-Q-ALOHA algorithm for a network of 100 nodes. The threshold policy clearly arises at the intersection of the two gradient columns.

do not represent a sharp threshold due to the different outcomes that each node experiences. In fact, looking for example at a sampled Q-matrix in the Figure 4.2, the agent start to transmit (with a proper probability τ) slightly before its threshold, and this is given by the randomness of the simulations. Going back to Figure 4.4, the darker vertical line represents the threshold of the reference policy for a network of 100 nodes, that we recall being equal to 220. Thus, a sub-optimal solution with respect to threshold-ALOHA policy is found by the AoI-Q-ALOHA. Such phenomena is tightly linked with the decentralized learning and the adaptive transmission probability: the agent does not know that it can do better than that by increasing its threshold *and* by being slightly more aggressive on the transmission probability. The lack of connection between the Q-matrix and the adaptive transmission probability (the second term learns because of the first one, not viceversa) leads the agent to rely on its interaction with the receiver, that we know being partially informative.

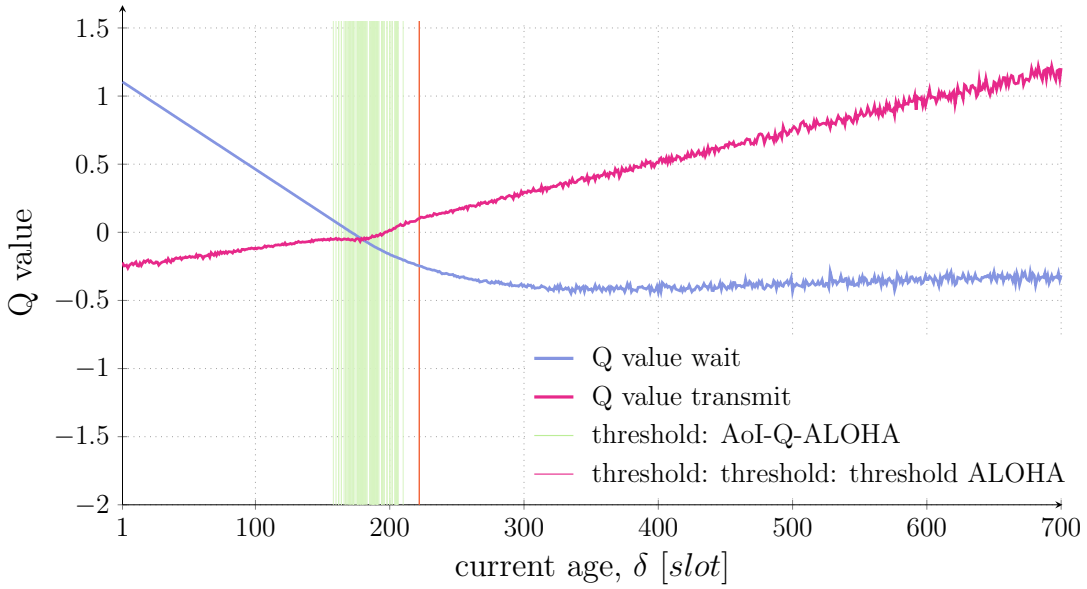

Figure 4.4: Average Q-matrix displayed as action value (y-axis) versus age (x-axis). Threshold policy is displayed as the two actions curve intersect. The vertical lines represent the retrieved thresholds from each agent, plus the theoretical threshold from threshold-ALOHA policy.

Transmission Probability

Finally, we consider how nodes adapt their transmission probability in AoI-Q-ALOHA. Since the algorithm converges in its RL component, we expect to find, after convergence, a steady transmission probability. Moreover, because the threshold retrieved by the agents is lower than the one presented in the benchmark policy, the transmission probability should reflect this result. Looking at the Figure 4.5, where the evolution over time of the mean adaptive transmission probability of the overall channel is displayed, a stable trend is showing just after few thousands of slots. For a network of 100 nodes, we know the optimal transmission probability, according to the threshold-ALOHA policy, being equal to ≈ 0.04, while the AoI-Q-ALOHA seems to converge at 0.03. To better understand this behavior, we shall remark that a generic threshold-policy in a SA framework, presents a direct proportionality between the transmission probability and the threshold: the higher the threshold, the higher the transmission probability. This is because by setting the threshold higher, fewer nodes will be active and so they transmission policy can get aggressive. A similar trend happens for lower thresholds: if the node has a relatively low threshold, then there is no need to be competitive within the channel and the transmission value can be reduced.
The curve is noisy because of the moving nature of the transmission updating rule: each time a transmission is performed, τ is updated. The parameter is modified smoothly, to avoid abrupt

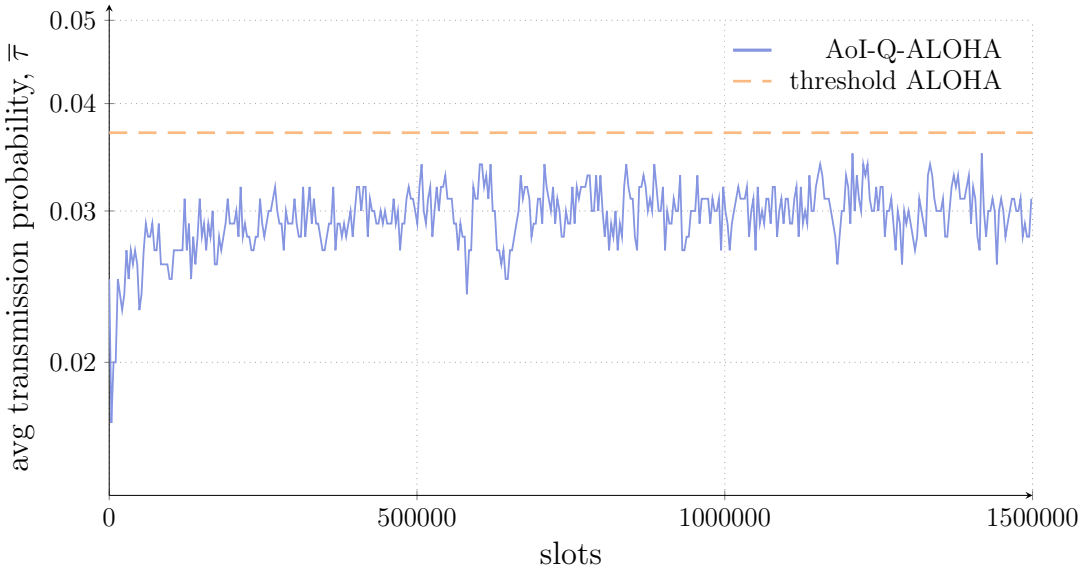

Figure 4.5: Transmission probability evolution throughout the simulation.

changes. This is because τ highly influences the *speed* of convergence of the algorithm, but not the results. Considering the probability bounded on both sides (upper and lower side) by reasonable values (i.e. so that the node won't get access probabilities so high that an exclusive usage of the channel is done, while the others nodes are getting shut down; and viceversa) then the updating rule should stabilize when the number of collisions and successes are true to a standard SA protocol. The upper and lower limit of the transmission probability is large enough to remove any assumption on the network topology. For a network of 100 nodes, we had the following constraint:

$$\tau_t \in [0.001, 0.1] \quad \forall t$$

where 0.001 is the standard transmission probability of SA network of 1000 nodes, including in the procedure a broad range of network sizes. In this way, any statistics about the number of devices is unnecessary, making the algorithm suitable for a real-world application.

4.1 Hyper-Parameters influence

We now continue with studying the influence of the hyper-parameters. By keeping fixed the setup except for the parameter of interest, we dig into the relationship between the learning process and the role of the specific term. We shall recall the main hyper-parameters of a RL-based algorithm are the *learning rate* α, the *discount factor* γ and the *exploration rate* ϵ.

4.1.1 On the learning rate

The learning rate, α, is the term quantifying the amount of fresh knowledge that is going to be integrated into the Q-matrix. If the agent is near convergence, a higher learning rate will speed up the process; if the agent is far away from the solution, then a lower learning rate will put the agent in a recoverable situation. Thus, the learning rate needs to be setup around a trade off of such situations. Generically speaking, the standard value assigned is 0.1. In this section, we present the results at convergence of the channel AoI using different learning rate throughout the simulation. Specifically, we tested the following values:

$$\alpha \in \{0.005, 0.1, 0.2, 0.3, 0.5\}$$

In Figure 4.6 the normalized mean AoI of the channel versus the number of slots is displayed per different α values. Surprisingly, for the majority of the terms there is no significant change

in the AoI trend. More specifically, for $\alpha \leq 0.3$ the minimization process follows the same curve: roughly speaking, the simulations converge to the same value that is $\approx 1.7\ [\frac{slots}{node}]$. The only term that emerges with a shaky trend is the learning rate of value 0.5. By setting the speed convergence at that high rate, we simply encounter the well know phenomenon of overshooting, a typical concern in gradient descent methods, where the algorithm jumps around the solution without actually converging to it. The resulting pattern is an oscillatory curve.

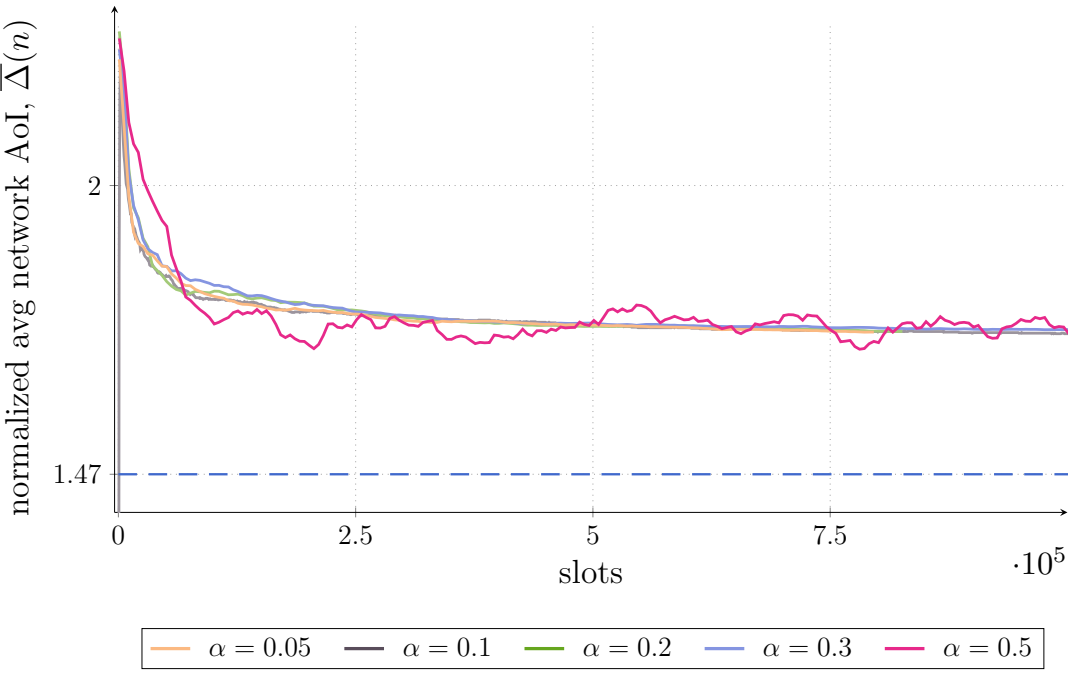

Figure 4.6: Normalized mean AoI against number of slots per different learning rate values.

4.1.2 On the discount factor

While the learning rate is common to most learning algorithms, the discount factor belongs to the Q-learning method. The discount factor defines how much the agent is going to relate on the rewards in the distant future rather than the immediate rewards. For a discount factor $\gamma = 0$ we say the agent is myopic and takes into account only the immediate actions and rewards. For a discount factor $\gamma = 1$, the agent gives the same weight to all of the actions taken throughout the simulation. As for the learning rate, the standard value used in our simulation

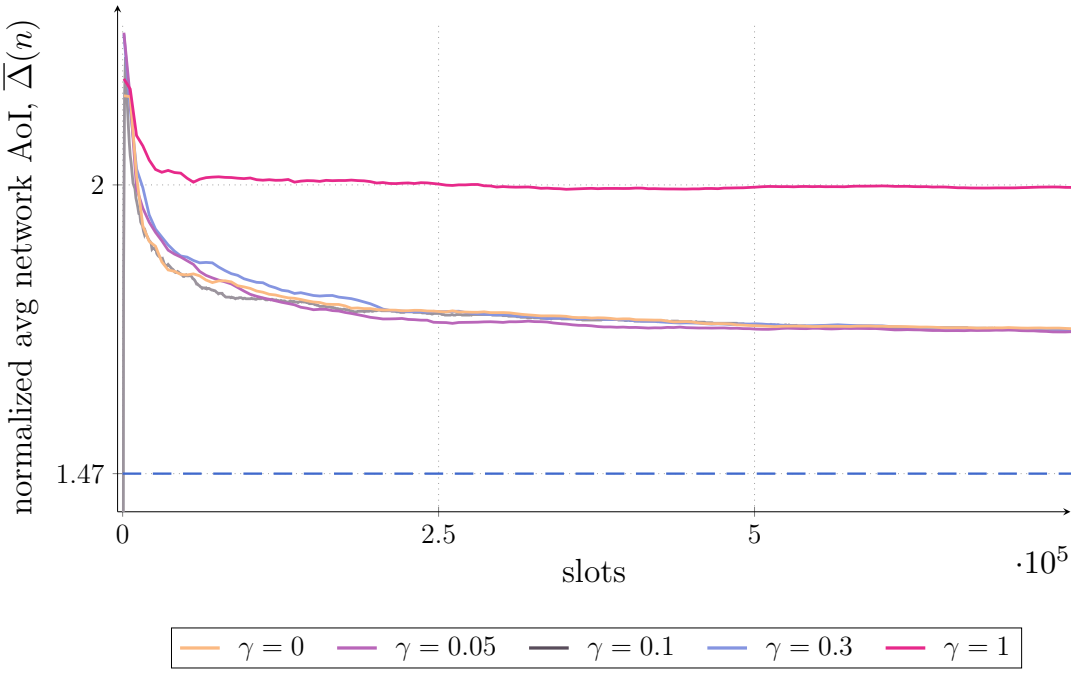

Figure 4.7: Normalized mean AoI against number of slots per different discount factors values.

is 0.1. For this section we tested several discount factors:

$$\gamma \in \{0, 0.05, 0.3, 1\}.$$

In Figure 4.7 the mean channel AoI versus number of slots of different discount factors is shown. A clear distinction can be observed: for an agent that takes into account every interaction at the same level ($\gamma = 1$), we have convergence at a significantly far away value if compared to the others. By setting the discount factor as the highest possible, we include into the Q-matrix all possible interactions, of which a subset of them are clearly not important due to exploration phases or randomness. As a consequence, because every outcome is taken at the same level of importance, convergence is reached but not at desirable values. As to the other configurations, no major differences are showing, even for $\gamma = 0$. Given the framework of just two actions, merged to a quite repetitive environment, no complex dependencies between the action taken and its future emerge. Because the node will repeat the same course of actions and state for ever, it has no need to project its policy onto the future as it will do it naturally by gaining experience over time. Such outcome is strictly related to the nature of the slotted ALOHA protocol. Given that, the discount factor does not play a key role in this kind of application.

4.1.3 On the exploration rate

Finally, we present the results related to the influence of the exploration rate. In Figure 4.8, the mean channel AoI trend is shown, at different exploration rates. Specifically, the values tested are the following:

$$\epsilon \in \{0, 0.01, 0.05, 0.1\}.$$

The plot shows how the longer the exploration rate, the higher the resulting mean AoI. Among the presented parameters, the best choice would be $\epsilon = 0$. The result is not surprisingly. The main reason lies in the nature of the algorithm: each time a node chooses to transmit, it does with a tailored transmission probability, instead of acting deterministically. As a consequence, the use of a probability adds to the agent a significant portion of exploration rate. Thinking about the first phase of learning, the node has no knowledge about the proper transmission probability and so by being in the wrong space will help the agent to explore enough states to satisfy the Q-learning convergence theorem. Thinking about the convergence instead, the exploration of states decreases as the policy gets improved. Thus, the exploration rate parameter is already integrated into the proposed algorithm.

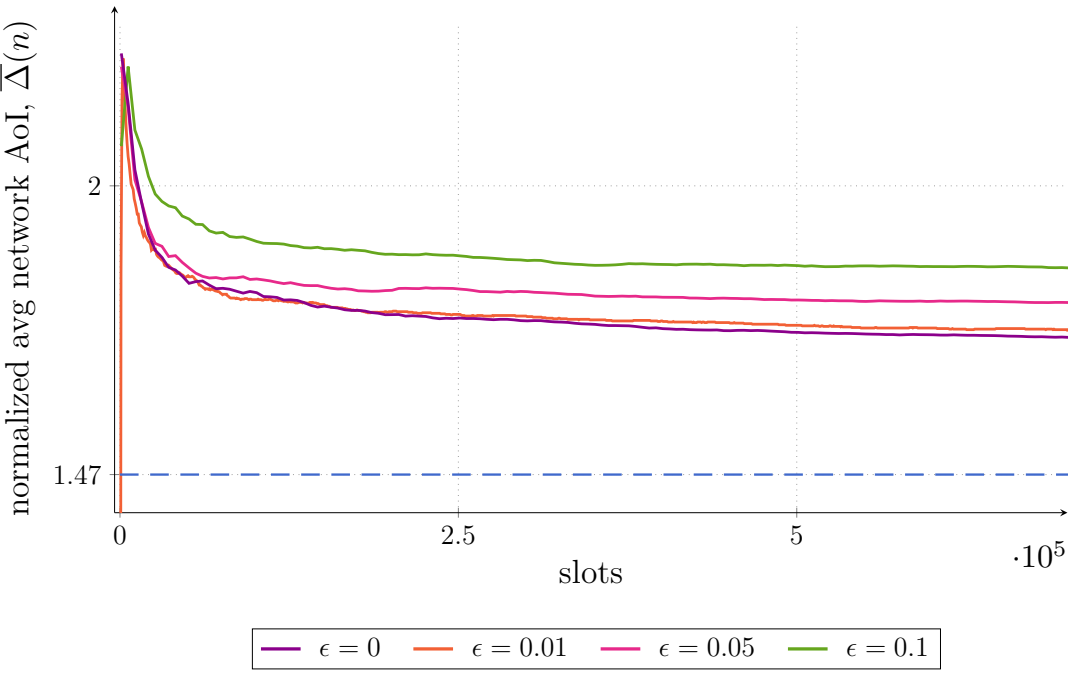

Figure 4.8: Normalized mean AoI against number of slots per different exploration rates.

4.2 Network Size

So far, we have presented the main features of the AoI-Q-ALOHA algorithm on a fixed network of 100 nodes. We continue our study by observing the mean AoI at convergence per different network sizes. In Figure 4.9, we report a comparison between the simulations of the threshold-ALOHA policy and the AoI-Q-ALOHA for network sizes ranging from 50 to 250. A decreasing trend is clearly showing as the number of nodes increases: from a size number of 50 up to 250 the final AoI is decreased by the 9%, leading to promising results for massive networks. The reasoning behind such behavior partially belongs to the intrinsic nature of Q-learning. By being a deterministic algorithm, a smaller network is more likely to fall within a "pre-ordinated" allocation of the channel, in the sense that a subset of nodes (if not the totality) will get by chance an age-state in which it is going to transmit while the others are silent. In other words, the Q-matrix at convergence will be noisy enough to prefer to transmit early than their mean-based threshold. However, if the nodes are few, then the transmission probability increases and so the chances of encountering collisions. As a final result, the nodes tend to collide more often and so their mean AoI to increase and decrease over time, leading to a worse performance. As the number of nodes increases, the Q-matrix tend to get less noisy and so the final policy more uniform among nodes. Moreover, we shall remark that all simulations have

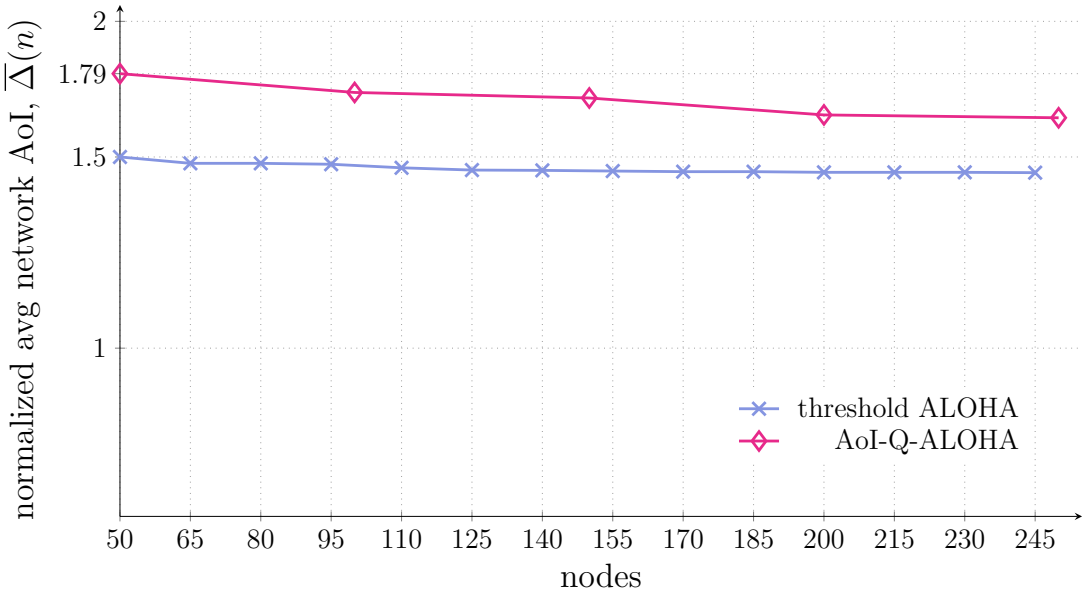

Figure 4.9: Mean AoI at convergence per different network sizes.

been held with the same exploration rate $\epsilon = 0.01$, and so not normalized with respect to the number of total nodes. It is clear that choosing a random action once every 100 slots is not the same for a network of 250 nodes and for a network of 70 nodes.

Moreover, another interesting insight that could explain the resulting AoI for smaller networks can be deducted from Figure 4.10, where two trained Q-matrices of a 70-nodes network are shown. As for Figure 4.2, the white cells represent the actions that the agent is going to pick for the specific age/row, without considering exploration phases. The two sampled nodes display different thresholds: around 100 and 150 of age, respectively. In a threshold-based SA policy, each active node is going to interfere with a number of active nodes that is smaller than the actual network size as some of them are temporarily silent. As a consequence, the number of interferences among the agents will make the node's mean age dependent on such value. In our case, the threshold is based on the node's mean age, and so the retrieved policies are representative of two different network states. By having a variable sub-network of active users, the agent may jump from one policy to another, and so increasing and decreasing the mean AoI. Such phenomenon tend to fade as the number of users increases, as we can see from the vertical green lines in Figure 4.4; bringing evidence to the fact that smaller networks converge at higher AoI values than the bigger ones.

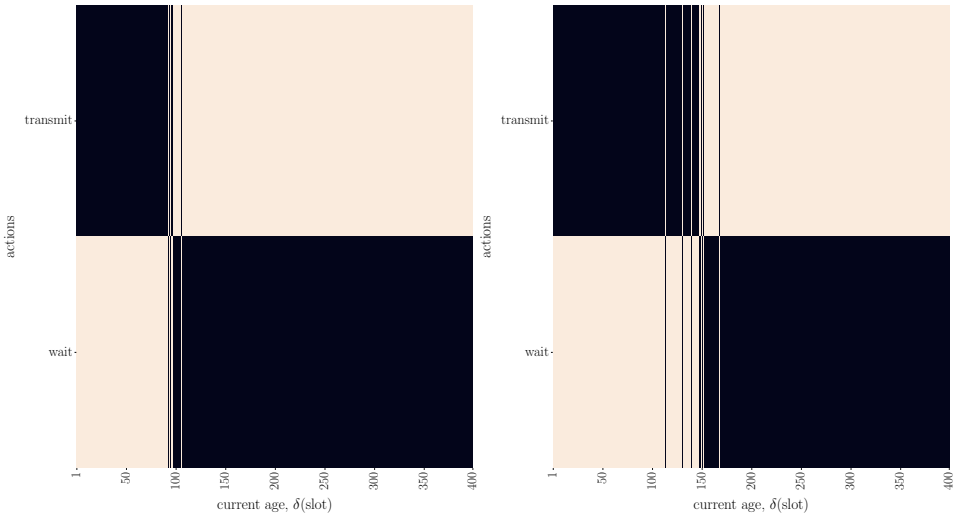

Figure 4.10: Two different policies of the AoI-Q-ALOHA for a network of 70 nodes.

4.2.1 Convergence Time

Another interesting insight that can be deduced from the AoI-Q-ALOHA is the convergence time per different network sizes. In order to present a proper differentiation on the values, we considered two scenarios with 70 and 250 nodes. In Figure 4.11, the normalized average channel AoI of the two networks are displayed during the first 3×10^5 learning slots. As expected, the smaller network converges quicker. The intuition behind relies on the fact that a smaller network requires less collisions and successes for the nodes to develop a policy. An user of a 70-nodes channel will stabilize around a threshold of 150 slots, while in a 250-nodes channel, the threshold at convergence will be 550 slots. Roughly speaking, the bigger network finds convergence 3 times slower than the smaller one. Such ratio is vaguely correlated to the network size ratio. Moreover, we shall remark that the two configurations show different values of AoI at convergence for the reasons explained in the previous section. The bigger the network, the lower the AoI.

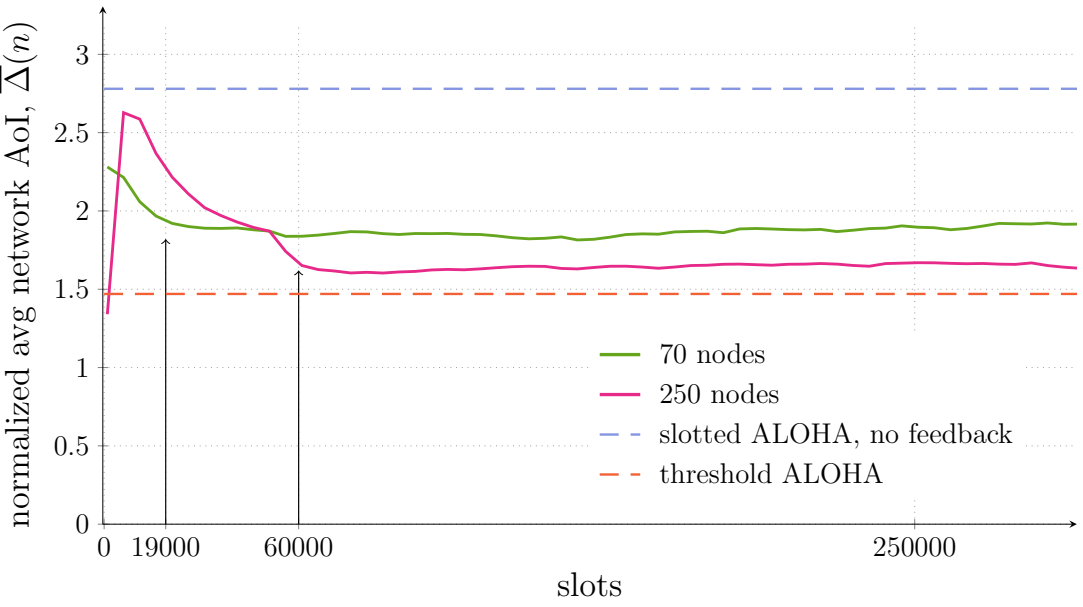

Figure 4.11: AoI trend per two different network sizes.

4.3 Variable Network

To conclude our discussion, we studied the robustness of the algorithm to network dynamics. In particular, we discuss the results on how the agents adapt their policy when the number of users change over time. We note that this aspect is especially relevant in IoT systems, where new nodes may join the network, and others may leave it due to battery adaption, for example. From this standpoint, while the knowledge of the current population size can be extremely useful (like in the case of threshold-ALOHA), it may not be available to nodes in practical settings. A major strength of the approach we propose lies exactly n is not requiring such information. To test this, we consider a simulation in which the network starts with 100 nodes, while so 50 additional nodes are later on integrated into the network, without warning the current users. What we expect from the AoI-Q-ALOHA method is a an adaptation from the old nodes to the new configuration. By adding a subset of users, the ratio of collisions should increase and so their policy must change as a response. Once the agents are stabilized into the new setting, a subset of 50 nodes are subsequently removed from the network. Once more, the remaining nodes should re-adapt their policy to the modified channel. We thus differentiate two major moments within the simulation: a *switch on* phase, where new users are integrated, and a subsequent *switch off* phase, where a subset of users are removed from the environment. We remark

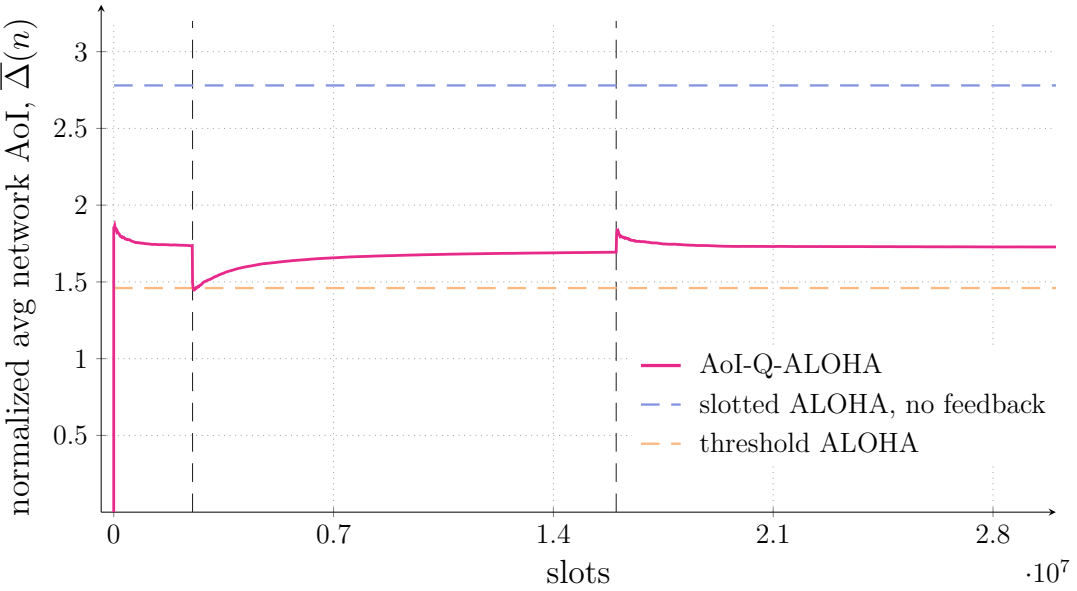

Figure 4.12: Normalized AoI for a variable network size.

that the latter phase predicts to remove a random subset of nodes, and not necessarily the same nodes that were being introduced into the starting configuration. In Figure 4.12, the running average AoI of the channel is shown, during all of the presented setting variations. The dashed vertical lines represents the starting points of the switch on and switch off phase, as it is clearly understandable from the change of the AoI trend. Notably, the multi-agent network quickly adapts to the variations. Moreover, the AoI at convergence reflects the results discussed in the previous section: when the network gets bigger, the final AoI should gets lower. In fact, in the switch on phase -$(100 \rightarrow 150)$ - the normalized AoI converges at $1.69^{f}racslotnode$ while in the subsequent switch off phase -$(150 \rightarrow 100)$ - converges to $1.72\frac{slot}{node}$.
The integration of new nodes is surely the crucial aspect of such simulation. To dig more into it, a plot of the AoI behavior exhibited by the two subset of nodes are displayed in Figure 4.13. The dark coloured line tracks the mean AoI of the older nodes, the one with whom the simulation has started; with the light coloured line instead, the recently inserted nodes AoI is tracked. The dashed blue line is the mean AoI returned by the algorithm during the simulation. In order to understand the reason of the gap between the curves, it becomes necessary to observe that the newest nodes do not have the same opportunity to learn as of the remaining nodes given that they already have a tailored policy to access the channel. In such a way, the optimization of their mean AoI is done "from above" and not "from below" given the imbalance on the

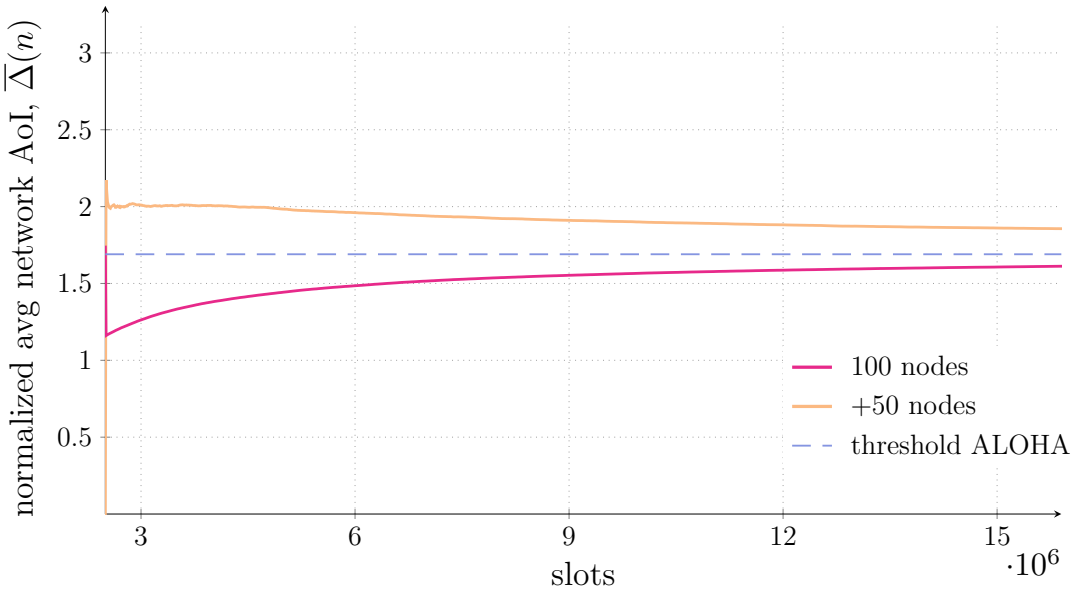

Figure 4.13: Trend of mean AoI of the old and new nodes into the shared channel.

number of collisions between the new and old nodes. In conclusion, a difference between the two subsets of nodes is going to be part of the final results. The already trained nodes have an advantage over the newer ones. We shall remark that the inability of the agents to communicate between each other worsen up scenarios like this. Nonetheless, the ability of AoI-Q-ALOHA to reset to such changes, together with its simplicity, prompts it as a very promising solution for IoT systems.

5 Conclusions

In this thesis, we designed and evaluated the performance of a Multi-Agent RL-based algorithm, the AoI-Q-ALOHA, aiming to improve the mean AoI of a SA protocol in the Remote Monitoring scenarios for IoT. The timeliness of nodes with respect to the receiver is one of the main concern when talking about tracking procedures. With the recent rise of Reinforcement Learning, a plethora of methods, usually merged with Deep Learning architectures, have been implemented on several SA variants to keep the mean AoI relatively low. However, the majority of results proposed in literature required the agents to be aware of network insights such as the number of devices; which is something that is often lacking in generic IoTs frameworks. Thus, the goal of this thesis has been the optimization of mean AoI within a decentralized setting. By keeping the communication scheme among users true to reality, this algorithm has been shaped around a practical implementation.

The AoI-Q-ALOHA is a distributed Q-learning algorithm where each agent is able to learn an access channel strategy solely based on its interactions with the receiver and through the acknowledgment, which is kept by default in feedback-based networks so that the running average age is tracked by each user. The nodes, by collecting *local* rewards, are able to develop AoI-based policies on how and when to access the channels. The reward has been shaped on a threshold policy where the node remains silent until a certain age, and starts to contend for the channel with a certain probability above that. The integration of an adaptive transmission probability had the intention to adjust a deterministic method, such as the Q-learning, to a probabilistic environment, such as SA. The objective, that is the minimization of the mean

channel AoI, requires an implicit cooperation among the users as the threshold and the transmission probabilities are based on the number of devices within the network, something that each agent is unaware of. Indeed, each agent acknowledges the receiver as the whole environment. To let the node adapt to a shared channel, the ACK signal is used as the main global incentive during the learning process.
The results reported in Chapter 4 show remarkable improvements with respect to standard slotted ALOHA performances reducing the average normalized AoI from a mean value of $\approx 2.7 \frac{\text{slots}}{\text{nodes}}$ to $\approx 1.7 \frac{\text{slots}}{\text{nodes}}$ for a network of 100 nodes. Further improvements are seen as the network size grows. Each agent developed a threshold policy based on the node's mean age i.e. the threshold is around its individual running average AoI. Alongside the threshold, an adaptive transmission probability of ≈ 0.03 for a network size of 100 has been adopted throughout the simulation; significantly higher than the one maximizing throughput in the SA channel, i.e. $0.01 = \frac{1}{100}$. Such tests have been extended to different network sizes: from shallow networks of 50 nodes up to bigger ones like 250. The deterministic nature of the Q-learning, merged with a significant high transmission probability (if compared to bigger networks) tend to let the nodes concentrate around two different active states (i.e. above threshold). Such pattern tend to dissolve for bigger networks, and so the -implied- cooperation among users improves.
Furthermore, the algorithm robustness has been tested on a dynamic network, where the number of nodes changes over time, demonstrating significant results in terms of policy shaping and mean channel AoI. Indeed, the nodes are able to adapt over time to a changing environment, by showing little discrepancies between the old and new nodes's average AoI at convergence. Thus, each agent is able to change their policy without being aware of the shift on the network cardinality, but by only interacting with the common receiver. Along the work, a comparison with benchmark threshold-based policy, threshold-ALOHA [12], has been carried out. The final results show a sub-optimal solution if compared to the cited work. Nonetheless, the threshold-policy has been proposed with the knowledge of the number of nodes at priori. The AoI-Q-ALOHA algorithm got inspired by the cited work and tried to overcome the share of information among users by the paradigm of Reinforcement Learning.

5.1 Future Works

One of the main reasons why the Q-learning has been chosen in this thesis is because of the small dimensionality of the state-action space and its quick convergence pattern. Indeed, the number of actions considered in this work is just two, and the results discussed in the Chapter

4 clearly show a fast convergence just after few thousands of slots. However, Q-learning is a deterministic method that chooses the best action for each state, and to overcome such flaw, we have combined it with an adaptive transmission probability that follows the learning process but surely has an influence on the complexity of the intelligent system. To overcome this, the action state space could be changed to a probabilistic one: instead of using deterministic actions (i.e. wait or transmit), a set of transmission probabilities could be used. In that case, the outcome of each action is probabilistic and so a tailored reward should be designed for it. On the other hand, since the action space gets complex, on-policy algorithms [15] usually overcome the curse of dimensionality by directly searching in the policy-space instead of approximating the optimal strategy through the action-state space. Family of algorithms like REINFORCE [37] and Actor-Critic methods [38], look for the optimal policy by estimating the gradient of returns. Moreover, other MDPs frameworks like Average-Payoff algorithms are meant to maximize the average payoff instead of the discounted long term reward. This setup comes at help when the nature of the problem is cyclic, like in SA: for each slot, the agents contend the same limited bandwidth on an infinite horizon of events. R-learning [39] is a well-known method of this family of MDPs problems.

Deep Learning architectures [40] could be stacked up in the learning process to help the approximation. Many Deep Reinforcement Learning algorithms [34] are now gaining attention thanks to their ability on handling high-dimensional spaces. Usually coupled up with on-policy algorithms, methods from the Actor-Critic family are recently being tested for large-scale scenarios [41]. The Proximal Policy Optimization (PPO) algorithm [42], is a sub-class of the Actor-Critic family which try to overcome the shortcoming of Q-learning. In the work of [33], a PPO-based method is used to develop medium-access strategies based on the current AoI and the urgency to transmit into the channel. Generally speaking, nonetheless the outstanding outcomes present in literature, in order to implement a practical-oriented work, the feasibility in terms of costs and computational resource should be studied alongside.

In conclusion, this work has shown how a decentralized multi-agent Q-learning paradigm was capable of optimizing a joint objective without the communication between the agents but solely with local interactions [43]. In spite of the unawareness of the presence of other agents, each single policy has been shaped so that every agent has a fair chance of transmitting into the channel and so by resetting their age. The presented work has shown remarkable results with the only usage of tabular methods, leaving the future work to be extended to a more complex system where novel architectures could be applied on.

References

[1] K. Rose, S. Eldridge, and L. Chapin, *The Internet of Things (IoT): An Overview – Understanding the Issues and Challenges of a More Connected World*. Internet Society, 2015.

[2] J. Luo, Y. Chen, K. Tang, and J. Luo, "Remote monitoring information system and its applications based on the Internet of Things," in *international conference on future biomedical information engineering (FBIE)*, December 2009.

[3] Y.-F. Kung, S.-W. Liou, G.-Z. Qiu, B.-C. Zu, Z.-H. Wang, and G.-J. Jong, "Home monitoring system based internet of things," in *IEEE International Conference on Applied System Invention (ICASI)*, April 2018.

[4] F. Fernandez and G. C. Pallis, "Opportunities and challenges of the Internet of Things for healthcare: Systems engineering perspective," in *4th international conference on wireless mobile communication and healthcare-transforming healthcare through innovations in mobile and wireless technologies (MOBIHEALTH)*, November 2014.

[5] N. Abramson, "THE ALOHA SYSTEM: another alternative for computer communications," in *Fall Joint Computer Conference*, November 1970.

[6] ——, "The Throughput of Packet Broadcasting Channels," *Communications, IEEE Transactions on*, vol. 25, no. 1, pp. 117 – 128, 1977.

[7] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of Information: An Introduction and Survey," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1183–1210, 2021.

[8] L. Badia and A. Munari, "A game theoretic approach to age of information in modern random access systems," in *Proc. IEEE Globecom Workshops*, December 2021.

[9] D. Bertsekas and R. Gallager, *Data networks (2nd ed.)*. Prentice-Hall, Inc., 1992.

[10] R. Yates and S. Kaul, "Age of Information in Uncoordinated Unslotted Updating," in *2020 IEEE International Symposium on Information Theory (ISIT)*, February 2020.

[11] A. Munari, "Modern Random Access: An Age of Information Perspective on Irregular Repetition Slotted ALOHA," *IEEE Transactions on Communications*, vol. 69, no. 6, pp. 3572–3585, 2021.

[12] O. T. Yavascan and E. Uysal, "Analysis of Slotted ALOHA With an Age Threshold," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1456–1470, 2021.

[13] R. Bellman, "Dynamic Programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[14] R. Nian, J. Liu, and B. Huang, "A review On reinforcement learning: Introduction and applications in industrial process control," *Computers Chemical Engineering*, vol. 139, p. 106886, 2020.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. A Bradford Book, 2018.

[16] S. V. Albrecht, F. Christianos, and L. Schäfer, *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024.

[17] M. Bowling and M. Veloso, "An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning," August 2001.

[18] J. Jiang, K. Su, and Z. Lu, "Fully Decentralized Cooperative Multi-Agent Reinforcement Learning: A Survey," 2024.

[19] H. Mao, Z. Gong, and Z. Xiao, "Reward Design in Cooperative Multi-agent Reinforcement Learning for Packet Routing," *CoRR*, vol. abs/2003.03433, 2020.

[20] K. Zhang, Z. Yang, and T. Başar, "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms," *ArXiv*, 2019.

[21] Y. Shoham, R. Powers, and T. Grenager, "Multi-Agent Reinforcement Learning:a critical survey," June 2003.

[22] R. D. Yates and S. K. Kaul, "Status updates over unreliable multiaccess channels," in *IEEE International Symposium on Information Theory (ISIT)*, May 2017.

[23] R. Talak, S. Karaman, and E. Modiano, "Distributed Scheduling Algorithms for Optimizing Information Freshness in Wireless Networks," in *IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, June 2018.

[24] L. Badia, A. Zanella, and M. Zorzi, "A Game of Ages for Slotted ALOHA With Capture," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4878–4889, 2024.

[25] P. Hegde, L. Badia, and A. Munari, "Age of information for remote sensing with uncoordinated finite-horizon access," *ICT Express*, 2024.

[26] E. Đokanović, A. Munari, and L. Badia, "Distributed Access by Multiple Sources for Age of Information Minimization Over a Finite Horizon," 2024.

[27] L. Badia, "Age of information from two strategic sources analyzed via game theory," in *Proc. IEEE CAMAD*, 2021.

[28] L. Crosara, M. Brocco, C. Cavalagli, X. Wu, E. Gindullina, and L. Badia, "Data Injection in a Vehicular Network Framed Within a Game Theoretic Analysis," in *Proc. IEEE MedComNet*, 2023, pp. 25–28.

[29] G. M. F. Silva and T. Abrão, "Throughput and latency in the distributed Q-learning random access mMTC networks," *Computer Networks*, vol. 206, p. 108787, 2022.

[30] S. Kosunalp, P. Mitchell, D. Grace, and T. Clarke, "Practical Implementation and Stability Analysis of ALOHA-Q for Wireless Sensor Networks," *Etri Journal*, vol. 38, pp. 911–921, 10 2016.

[31] E. T. Ceran, D. Gunduz, and A. Gyorgy, "A Reinforcement Learning Approach to Age of Information in Multi-User Networks with HARQ," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2021.

[32] M. A. Abd-Elmagid, H. S. Dhillon, and N. Pappas, "A Reinforcement Learning Framework for Optimizing Age of Information in RF-Powered Communication Systems," *IEEE Transactions on Communications*, vol. 68, no. 8, pp. 4747–4760, 2020.

[33] M. Jeong, G. Seo, and E. Hwang, "Age of Information Optimization by Deep Reinforcement Learning for Random Access in Machine Type Communication," in *2022 IEEE International Conference on Big Data (Big Data)*, December 2022.

[34] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[35] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A Theoretical Analysis of Deep Q-learning," vol. 120, 2020, pp. 1–4.

[36] M. Bowling and M. Veloso, "Rational and Convergent Learning in Stochastic Games," *IJCAI International Joint Conference on Artificial Intelligence*, 08 2001.

[37] J. Zhang, J. Kim, B. O'Donoghue, and S. Boyd, "Sample efficient reinforcement learning with REINFORCE," in *Proceedings of the AAAI conference on artificial intelligence*, 2021.

[38] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, vol. 12, 1999.

[39] S. Singh, "Reinforcement Learning Algorithms for Average-Payoff Markovian Decision Processes," *Proceedings of the AAAI Conference on Artificial Intelligence*, no. 12, 1999.

[40] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[41] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski *et al.*, "What matters for on-policy deep actor-critic methods? a large-scale study," in *International conference on learning representations*, 2021.

[42] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[43] C. Cavalagli, L. Badia, and A. Munari, "Reinforcement learning for age of information aware transmission policies in slotted ALOHA channels," in *Proceedings IEEE International Symposium on Wireless Communication Systems*, 2024.

Acknowledgments

Dedico questa sezione a tutte le persone che hanno incrocriato il mio cammino in questi anni di vita. E' difficile riassumere un arco di tempo così ampio, ma voglio ringraziare tutti quelli che, a modo loro, hanno lasciato qualcosa che porterò per sempre dentro di me.
Prima di tutto ringrazio il mio relatore Leonardo, per avermi dato l'opportunità di lavorare in un ambiente prestigioso e stimolante come DLR. Ti ringrazio per la fiducia data, per la tua simpatia e per la tua enorme disponibilità. Grazie al tuo sostegno ho iniziato a credere di più in me stessa e nel lavoro che stavo portando avanti. Grazie davvero.
Ringrazio Andrea, che come Leonardo mi è stato accanto in questo periodo di lavoro a DLR, e che con la sua gentilezza e disponibilità mi ha dato modo di poter scambiare idee e progetti insieme a lui. Lavorare con te è stato stimolante dal primo all'ultimo giorno, mi sono sentita parte di qualcosa, ed è grazie a te che ho conosciuto da vicino qual è il vero piacere della ricerca. Grazie per la fiducia e per il sostegno, soprattutto durante gli ultimi mesi dove il futuro spaventava più del presente. Grazie davvero.
Questa tesi è anche il frutto di tre anni di vita spesi tra una città che lentamente è diventata una seconda casa, Padova, ed un'altra che lascerà per sempre il segno dentro di me, Monaco di Baviera.
Vorrei quindi iniziare a parlare dall'inizio, dalla piccola Chiara che si è allontana da un piccolo paesino umbro per provare quella vita in città che tanto bramava. Non è stato facile, come ogni esperienza d'altronde. Dopo un anno di coinquilini e amicizie sfortunate, sono stata accolta in quella che è diventata la mia seconda casa, a cui sarò per sempre grata per avermi regalato momenti di quotidianità e spensieratezza. Ringrazio quindi Cinzia, Cristian e Lorenzo, per aver fatto parte del mio "ogni giorno" tra vicini di casa discutibilmente abili in cucina, e tra cene e pranzi condivisi insieme. Ringrazio in particolare Lorenzo, che oltre ad esser diventato il miglior coinquilino di sempre, è stato anche il mio compagno di triennale e magistrale; una presenza costante nella mia vita a cui sarò sempre grata per la vera amicizia che ci lega, e l'intesa che mi regala ogni volta memorabili risate. Ti voglio bene, davvero.
Ringrazio tutte le persone che hanno fatto parte del mio percorso di studi a Padova, in particolare Flavia: per esserci stata dal primo giorno, nonostante la distanza, nonostante le nostre vite differenti. Ti ringrazio per i momenti passati insieme, per le chiacchierate a Prato della Valle,

per i giri in bicicletta. Ringrazio tutti i miei amici per avermi fatto vivere Padova in tutte le sue sfumature.
Ringrazio Lorenzo e Martina, per essere stati i miei compagni di serate. Spero, un giorno, di poter riprendere la bici, ed andare a far serata con voi al Pride Village, o di potervi fotografare all'Anima Underground, come i vecchi tempi. Siete pazzi, vi voglio bene.
Ringrazio poi la Fotografia. Per me Padova rappresenta il momento di vita in cui la fotografia ha iniziato a farne parte. Chi mi è stato vicino da sempre sa che in questi ultimi due anni ho scoperto di avere un fuoco, quell'urgenza di dover dire qualcosa, di trasmettere quello che mi premeva dentro, con la fotografia. Ho sempre vissuto l'università come un dovere, come un disegno già prestabilito, ma mai come un piacere. La fotografia mi ha aiutato a scoprire me stessa e ad apprezzare giorno dopo giorno la vita che mi stavo lentamente costruendo, dandomi sicurezza ed autostima. E' quindi doveroso per me ringraziare tutte le persone fantastiche che ho conosciuto alla sede Irfoss a Padova, dove grazie a Riccardo ho scoperto il mondo cruento e passionale del fotogiornalismo e del documentariato. Ti ringrazio Riccardo per avermi spronata ad uscire dalla comfort zone, per avermi fatto sognare con tutte le tue storie, e per aver alimentato, con le tue parole gentili, quel fuoco dentro di me di cui non sapevo cosa farne fino a poco tempo fa. Te ne sarò per sempre grata, perchè grazie a te ho accettato e accolto la mia dualità tra scienza ed arte. Grazie veramente.
Saluto oggi Padova con il cuore in mano perchè sono cresciuta e cambiata tantissimo in questi tre anni. Me ne sono andata da quel paesino umbro detestando tutto ma oggi ritorno con la gratitudine di essere nata e cresciuta in un posto unico al mondo come Assisi. Ma che ne sapete!

E' ora doveroso ringraziare quella che è diventata un'altra tappa di vita di questo ultimo anno: Monaco. Lasciando Padova alle spalle, mi sono ritrovata in una nuova dimensione da un giorno all altro, senza nessuna aspettativa ma con la voglia di vivere.
Ringrazio prima di tutto Paul e Daniele, quelli che per me sono stati il mio "giorno 0" di questa esperienza. Terrò per sempre dentro di me la gentilezza di Paul, che mi ha accolto in casa dal primo momento e mi ha mostrato tutte le sfaccettature di Monaco. Ti ringrazio per quel legame che si è creato giorno dopo giorno, per il cuore onesto e gentile che hai, e per la tua compagnia. Ti voglio veramente bene, e ti auguro il meglio, in tutto.
Ringrazio poi Daniele, che da ex-compagno di magistrale è diventato il mio migliore amico in questi mesi difficili (ah si, anche divertenti) a Monaco. Mi piace pensare che è stato il destino a farci ri-incontrare, e come già sai, ti sono grata per tutto quello che abbiamo passato insieme. Ci siamo fatti da spalla dal primo all'ultimo giorno. Ci siamo ascoltati, ci siamo fatti compagnia

anche quando la solitudine sembrava essere l'unica persona presente nella stanza. Sono grata, e anche un po' fiera, di essere tua amica e di averti a fianco perchè sei una persona speciale. Ti voglio bene, davvero.
Sono fortunata di poter dire di aver conosciuto persone magnifiche a DLR, persone che non dimenticherò mai e a cui voglio un bene immenso. Ringrazio Salvatore, Luca, Davide, Jan, Iker, Luca, Matti, Thomas, Aurora, Matteo, Leo. Vi ringrazio per tutto, per i primi giorni di imbarazzo in aula studenti, le prime chiacchierate in S-bahn al ritorno dal lavoro, ai pomeriggi passati al parco a mangiare e bere insieme. E' difficile riassumere quello che si è provato e passato in così tanti mesi, ma voi sapete che vi terrò nel cuore per sempre. Tschüß!

Ed infine, ma non per importanza, ringrazio tutte quelle persone che mi piace definirle come pilastri, perchè ci sono da sempre, sono le fondamenta della mia piccola casa. Parto dalle mie amiche di scuola, quelle "storiche", quelle che dopo ormai dieci anni di amicizia te le devi tenere per forza, non ci sono altre soluzioni. Le ringrazio perchè nonostante la distanza e le poche occasioni, continuano ad essere un punto di riferimento, che sia una chiamata, un messaggio, o un uscita ogni quattro mesi: l'affetto è sempre quello. Ci ascoltiamo, ci siamo l'una per l'altra. Ve ne sarò per sempre grata, e anche un po' fiera, di avere queste persone accanto da così tanto tempo. Ringrazio quindi Alessandra, Laura, Lucia, Matilde, Chiara, Carolina, Sofia, Arianna, Fiore.
Ci sono poi altri tipi di amicizie, quelle che ti legano per il paese, per la scuola elementare, o chissà cosa. Quelle amicizie che ci sono e le coltivi senza sforzi, perchè fanno parte del tuo quotidiano ormai da troppo tempo. Ringrazio Vittoria, Costanza, Elena e Lucrezia, per essere una costante nella mia vita, in un modo o nell'altro. Sono contenta di potervi considerare ancora tali, nonostanti i momenti difficili, nonostante le nostre vite lontane e differenti. Siete un pezzo di me. Vi voglio bene.
Ringrazio in modo particolare Vittoria che è da sempre la mia migliore amica, e sempre lo sarà. Presente in tutto, allineate con il cuore e con la mente, ci accompagniamo in questa vita da non so quanti anni. Come già sai, ti voglio bene, e te ne vorrò per sempre. Grazie per essere sempre te stessa.
C'è infine un senso di gratitudine che è diverso dagli altri perchè forse è il più difficile da riconoscere ogni giorno. La fortuna di avere una famiglia che ti supporta e ascolta ogni giorno non è scontato. La fortuna di chiamare e di ricevere sempre risposta, che sia un momento bello o brutto, non è scontato. Ed è soprattutto nel lato distanza che Mamma forse mi ha sempre capito più di tutti, perchè lei ha provato le stesse cose. Ti ringrazio Mamma per volere sempre il

meglio per me, per accettare tutte quelle scelte che in fondo non condividi ma non ti fermano dal volermi bene. Ti ringrazio per tutto il tempo che dedichi a me quando ne ho bisogno, che sia una chiamata o una serata in casa. Ti ringrazio per il supporto e per cercare di aiutarmi ogni volta che ne hai l'occasione. Te ne sono grata e lo sarò per sempre, per quanto difficile sia ricordarselo ogni giorno.
Ringrazio poi Papà e mio fratello Paolo, che ci sono sempre stati nonostante la distanza e nonostante i nostri ritmi diversi. Ho sempre sentito il vostro calore e affetto anche da lontano, da un semplice chiedere "come stai" ad un messaggio di conforto durante i miei momenti buii. Sono cose che non dimentico, e che mi dimostrano quanto affetto sincero mi aspetta ogni volta a casa. Vi voglio bene.
Ringrazio in particolare mio fratello Paolo che con il tempo è diventato anche un amico, un posto sicuro dove poter parlare e sfogarsi. Sono contenta di essere tua sorella, fiera di starti accanto e di sapere di averti al mio fianco, per sempre. Non vedo l'ora di vedere cosa la vita ha in serbo per noi.
Ringrazio poi mia zia Tiziana e mia nonna Renata, che anche da lontano hanno sempre fatto sentire la loro presenza, con una chiamata o con un messaggio. Sento da sempre l'affetto che provate per me. Sono contenta di tornare a casa e di pranzare di nuovo con voi, come sempre. Vi voglio un bene immenso.
Infine, cugine di sangue ma amiche per scelta, ringrazio Cecilia, che continua ad essere una costante nonostante le vite incasinate e i problemi che i venti-cinque anni ci danno. Miglior compagna di viaggio, non vedo l'ora di riprendere un volo con te e di perderci in qualche città sconosciuta nel globo. Grazie per tutto, la tua presenza conta, ogni volta.
Grazie a tutte le persone che hanno incrociato il mio cammino e grazie a tutte quelle che lo faranno.
Grata per questa vita.