Decentralized spectrum learning for radio collision mitigation in ultra-dense IoT networks: LoRaWAN case study and experiments

Christophe Moy, Lilian Besson, Guillaume Delbarre, Laurent Toutain

▶ **To cite this version:**

Christophe Moy, Lilian Besson, Guillaume Delbarre, Laurent Toutain. Decentralized spectrum learning for radio collision mitigation in ultra-dense IoT networks: LoRaWAN case study and experiments. Annals of Telecommunications - annales des télécommunications, 2020, 75 (11-12), pp.711-727. 10.1007/s12243-020-00795-y . hal-02956350

HAL Id: hal-02956350
https://hal.science/hal-02956350

Submitted on 2 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Annals of Telecommunications
https://doi.org/10.1007/s12243-020-00795-y

Decentralized spectrum learning for radio collision mitigation in ultra-dense IoT networks: LoRaWAN case study and experiments

Christophe Moy[1] · Lilian Besson[2] · Guillaume Delbarre[1] · Laurent Toutain[3]

Received: 10 July 2019 / Accepted: 10 August 2020
© The Author(s) 2020

Abstract
This paper describes the theoretical principles and experimental results of reinforcement learning algorithms embedded into IoT devices (Internet of Things), in order to tackle the problem of radio collision mitigation in ISM unlicensed bands. Multi-armed bandit (MAB) learning algorithms are used here to improve both the IoT network capability to support the expected massive number of objects and the energetic autonomy of the IoT devices. We first illustrate the efficiency of the proposed approach in a proof-of-concept, based on USRP software radio platforms operating on real radio signals. It shows how collisions with other RF signals are diminished for IoT devices that use MAB learning. Then we describe the first implementation of such algorithms on LoRa devices operating in a real LoRaWAN network at 868 MHz. We named this solution IoTligent. IoTligent does not add neither processing overhead, so it can be run into the IoT devices, nor network overhead, so that it requires no change to LoRaWAN protocol. Real-life experiments done in a real LoRa network show that IoTligent devices' battery life can be extended by a factor of 2, in the scenarios we faced during our experiment. Finally we submit IoTligent devices to very constrained conditions that are expected in the future with the growing number of IoT devices, by generating an artificial IoT massive radio traffic in anechoic chamber. We show that IoTligent devices can cope with spectrum scarcity that will occur at that time in unlicensed bands.

Keywords Internet of Things (IoT) · Machine learning · MAB · Bandit · UCB · Radio spectrum · Collision mitigation · Interference · LoRa · Artificial intelligence · LoRaWAN · Cognitive radio · Spectrum scarcity · ISM band

1 Introduction

Wireless Internet of Things (IoT) is based on low-power wide-area networks (LPWAN) able to interconnect low-cost and mostly battery-powered devices over long ranges to an access point to the Internet. This is made possible by the use of low bit rates, low-bandwidth machine-to-machine (M2M) types communications. After the expansion of human-to-human mobile communications in the 1990s and then human to the Internet communications in the 2000s, now has come the era of M2M and especially Machine to the Internet (M2I). M2I is expected to experience a tremendous expansion in the very next few years, through IoT networks.

We can consider two categories of IoT networks. First are the cellular IoT networks, deployed by mobile phone operators, running 3GPP standards such as EC-GSM IoT, LTE-Cat0, LTE-Cat M1, NB-IoT, or expected 5G IoT. These standards will be supported in licensed frequency bands operated for cellular telephony. The second category of IoT wireless networks uses unlicensed bands for wireless links, also called ISM bands, which are open to the use for industrial, scientific, and medical applications. The most commonly used ISM bands are 434 MHz and 868 MHz in Europe and Africa and 915 MHz in America, with worldwide bands at 2.4 GHz and 5.8 GHz. Due to the constraints in terms of range and bandwidth, the 868 MHz and 915 MHz bands are mostly preferred for IoT networks. They communicate through protocols based on very different radio physical layer and medium access control specifications. For instance, the current two most well-known IoT standards are LoRaWAN [1], based on a chirp spread-spectrum solution, and Sigfox [2], based on an ultra-narrow band technology.

✉ Christophe Moy
christophe.moy@univ-rennes1.fr

[1] Univ Rennes, CNRS, IETR - UMR 6164, F-35000 Rennes, France

[2] CentraleSupélec, CNRS, IETR - UMR 6164, F-35576 Cesson-Sévigné, France

[3] IMT Atlantique, IRISA, F-35700 Rennes, France

Published online: 27 August 2020

In cellular licensed IoT networks, only one transmission may occur in a given place, at a given time, and in a given frequency band, between any operated device and the radio access point, scheduled by the cellular network. However, in unlicensed bands, IoT networks face very different and specific conditions. Many IoT networks can be deployed in the same area and overlap geographically, regardless if they are using the same protocol or not. Even if there exist rules to be followed in unlicensed bands, such as transmit power mask and duty cycle limits in the 434, 868, and 915 MHz band, for instance, many radio transmissions may collide at the same place, time, and frequency, as no global coordination is achieved.

The goal of this paper is to present the original *IoTligent* approach that embeds very low-cost machine learning algorithms inside IoT devices in order to make them smart. We named these IoT "intelligent devices": IoTligent [3]. Intelligence here is used in order to mitigate radio collisions and other jamming effects (propagation, malicious attacks, etc.) in the ISM bands. Low cost here is to be considered in terms of processing power, processing resources, memory footprint, protocol overhead, and frequency resources usage.

After exposing the issues we target in this work and the corresponding hypothesis in Section 2, Section 3 reminds the foundation of the learning algorithms used in *IoTligent*. Then, we show how we validated our approach through several gradual stages of experimentations. Measurements 1 of Section 4 give results of a proof-of-concept made in laboratory conditions using SDR (software-defined radio) platforms in order to validate the learning approach. Then, Section 5 gives the experimental architecture and hardware configuration used for measurement 2 campaign presented in Section 6. Experiments have been realized on LoRa IoT devices operated in real radio conditions of an operating LoRaWAN network in the city of Rennes (France). In Section 7, we present measurement 3 made in an anechoic chamber with an emulated radio traffic generator. We reproduce here the future very dense IoT networks radio conditions and validate the proposed learning approach for future ultra-dense LoRaWAN networks.

2 Collisions, hypothesis, and advantages of decentralization

2.1 Collisions vs. autonomy

Radio collisions are the main drawback for IoT in unlicensed band, both in terms of battery autonomy and also of IoT viability in the ISM bands itself. Indeed, collisions may cause (many) retransmissions at the cost of an increase of the RF (radio frequency) contention and may lead to a lower battery lifetime of the devices. Even worse, this could lead to a total failure of the IoT service, either because IoT devices cannot succeed in sending any data to the network or because multiple repetitions could make them consume all their energy much faster than expected.

2.2 Analysis of collisions

Radio collisions will be the weak point of LPWAN IoT networks operating in the unlicensed bands. Different kinds of collisions exist, as collision may occur with:

- Other IoT devices of the same network, as several networks covering the same area are not coordinated. This can occur between IoT devices uplink (UL) transmissions and between IoT UL and gateway downlink (DL) transmissions towards IoT devices.
- Other IoT devices of surrounding IoT networks using the same IoT standard. This can occur both in UL and DL, as surrounding IoT gateways of different networks are *not* coordinated. They could use the same channels or partly same and partly different channels.
- Other IoT radio signals using other IoT radio standards with different channels, bandwidth, users' repartition, etc.
- Other radio signals present in the ISM bands that are not IoT signals. By definition, they use completely different rules than IoT. They can be considered "jammers" from the IoT network point of view.

It is also important to note that, as each IoT standard uses its own rules for channeling and bandwidth, all this leads to an erratic spectrum usage, which cannot be planned, and has to be learnt in vivo. However, unlicensed band does not mean un-ruled band (there are for duty cycle, power, etc.), but they are more exposed to the non-respect of these few rules as regulation is relaxed and, thus, controls as well.

2.3 Other issues

Other issues can affect IoT transmission success. First one is propagation. Depending on the specific environment conditions around each device, it is unpredictable to know if radio propagation does affect all channels in the same way. Then, a channel facing bad propagation conditions would make IoT network to suffer from the same effect as made by collisions. Then electromagnetic circumstances could be disturbed in the area of devices, due to the proximity of other electric or electronic devices suffering from leakage radiations, as in factory environment, for instance. IoT signals may be very weak, and receivers should have a very low sensitivity in order to consume as less power as possible. Last but not least, malicious attacks could be made in the radio domain, but they could hardly cover all ISM bands and be permanent in order to stay discrete enough and undetected.

2.4 A device side solution for spectrum management

Our learning approach can help IoT devices to cope with all this kind of disturbances which act as jammers for the IoT spectrum in unlicensed bands. It imposes no change on the usual settings of IoT protocols, as, for instance, LoRaWAN [1]. It means that it imposes no extra retransmission, no data to be added in frames, no extra power consumption, etc. to be done. The only condition is that the proposed solution should work with the *acknowledged* (*ACK*) mode for IoT. The underlying hypothesis is that channels' jamming (even if there are no official channels in ISM bands) provoked by surrounding radio signals (IoT or not) and propagation effects is not equally balanced. In other words, some ISM sub-bands are less occupied or less jammed than others. However, it is not possible to predict it in time and space, by a centralized unit, so it has to be learnt on the fly, in a decentralized manner, i.e., on device's side.

The considered learning algorithms are a kind of artificial intelligence (AI) algorithms that are compatible with the constraint of low complexity of IoT devices, as we explain below. It is indeed much more efficient to implement a radio collision mitigation approach on the device side, as devices may be quite far away from gateways and suffer from different radio and jamming/co-existence conditions. But they are the place where every Watt counts at transmission and where sensitivity should be the best at reception, as no extra processing can be afforded.

2.5 Advantages of the proposed solution

The proposed approach is based on reinforcement learning algorithms studied in [4] and experimentally validated on real radio signals for cognitive radio and especially for opportunistic spectrum access (OSA) in [5]. We assert that, as for OSA, the IoT spectrum access issue can be modeled as a multi-armed bandit (MAB) problem [6] [7]. Reinforcement learning is based on a feedback loop that gives a success/failure measure of experience. In the IoT context, we propose to use the acknowledgement (ACK) sent by the gateway to the IoT device as a binary reward (1/0 for presence/absence of ACK). A reward of "1" means that a message has been successfully transmitted from the device to the gateway on the uplink channel, as well as the ACK message has been successfully transmitted from the gateway and received by the device on the same channel in downlink. A reward of "0" means that some collision, or other issue, happened either on uplink or downlink, so that ACK has not been received by the IoT device. Every device aims at maximizing its transmission success rate or, equivalently, at maximizing its cumulated reward (i.e., number of received ACK).

The main advantages of our solution are that:

- Bandit algorithms have strong mathematical proofs of convergence.
- Proofs are verified in real radio conditions, thanks to the good matching between models and reality.
- Learning converges very fast in real radio applications experiments [8].
- Implementation and execution both require very low processing and memory overhead, so that it is possible to add the proposed approach in IoT devices for a negligible complexity (processing, hardware, memory) and negligible extra energy consumption overhead.
- Learning can efficiently start from scratch, so there is no need for any prior training when deploying the IoT device (i.e., no need to lose some time to acquire this knowledge before operation really starts).
- Using such learning algorithms will never give worse results than a state-of-the-art random solution [9], even at the very beginning of the learning process, i.e., before learning brings a clear advantage.

Hence, we argue that the proposed approach can adapt to any kind of radio context, and we also note that:

- The stationarity of the environment is a requirement for the proofs of convergence, but if conditions change occasionally, convergence is so fast that a simple solution consists in resetting learning from time to time [9] (note that there also exists adaptive versions).
- No coordination is required between devices, but benefits decrease with the number of devices using the proposed solution, when it represents a great majority of devices (see the solutions presented in [9] [10]).
- As soon as a device is running in acknowledged mode, no overhead is added, neither in terms of protocol nor extra bits to be put into the LoRaWAN frames in uplink or downlink. A received ACK yields a reward of 1, and no ACK yields a reward of 0, without needing to change the content of the ACK messages.

3 MAB model and learning solutions

We model the IoT wireless spectrum issue as a MAB problem [6], and we propose to use bandit algorithms [7] at the IoT device side to solve this issue.

3.1 System model

We consider the system model presented in Fig. 1, where a set of devices sends uplink packets to the IoT network gateway.

The communications between IoT devices and the gateway are done through a simple ALOHA-based protocol, where

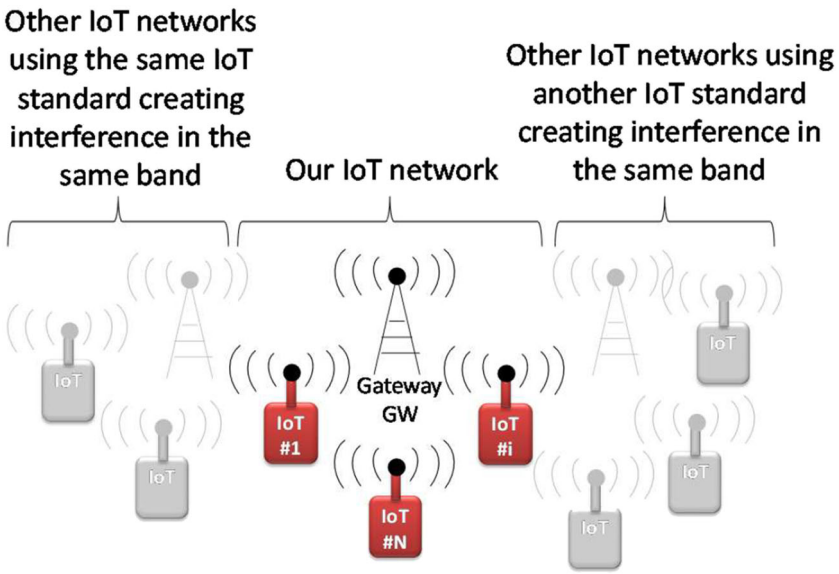

Fig. 1 System model used for IoT, with intelligent IoT devices that are able to dynamically set their transmission channel, thanks to a learning algorithm, in order to minimize collisions and interference from other radio signals in the unlicensed ISM band, especially other IoT networks which will be responsible of most of future traffic

devices transmit uplink packets of fixed duration, whenever they want. The devices can transmit their packets in one of the $K \geq 2$ channels. Channels are predefined (in frequency), but time is unslotted. In the case where the gateway receives an uplink message in one channel, it transmits an ACK response in downlink to the corresponding end device in the same channel, after a fixed delay. These communications operate in unlicensed ISM bands, and consequently, as stated in the previous section, they suffer in particular from interferences generated by uncoordinated neighboring networks. This interfering traffic is uncontrolled and can be unevenly distributed over the K different channels.

We consider the network from the point of view of a single IoT device. Every time slot has to communicate with the gateway (at each transmission $t \geq 1$, $t \in \mathbb{N}$), it has to choose one channel, denoted as $C(t) = k \in \{1, \ldots, K\}$. After transmission, the IoT device starts to wait in the same channel $C(t)$ for an ACK sent by the gateway. Before sending another message (i.e., at time $t + 1$), the IoT device knows if it received this ACK message or not. For this reason, selecting the channel (or *arm*) k at time t yields a (random) feedback, called a *reward*, $r_k(t) \in \{0, 1\}$, being 0 if no ACK was received after the previous message or 1 if ACK was successfully received. The goal of the IoT device is to minimize its packet loss ratio or, equivalently, to maximize its successful transmission rate, which here is its cumulative reward, as it is usually done in MAB problems [6] [7] [11]:

$$r_{1\ldots T} : = \sum_{t=1}^{T} r_{C(t)}(t) \tag{1}$$

This problem is a special case of the so-called "stochastic" MAB, where the sequence of rewards drawn from a given arm k is assumed to be *i.i.d.*, under some distribution ν_k, that has a mean μ_k. Several types of reward distributions have been considered in the literature, for example, distributions that belong to a one-dimensional exponential family (e.g., Gaussian, exponential, Poisson or Bernoulli distributions). As rewards are binary in our model, we consider only Bernoulli distributions, in which $r_k(t) \sim \text{Bern}(\mu_k)$, that is, $r_k(t) \in \{0, 1\}$ and $\mathbb{P}(r_k(t) = 1) = \mu_k \in [0, 1]$. Contrary to many previous works done in the cognitive radio field (for instance, in opportunistic spectrum access [12]), the reward $r_k(t)$ does *not* come from a sensing phase before sending the t-th message, as it would do for any "listen-before-talk" model. Rewards come from receiving an ACK from the gateway, between the t-th and $t + 1$-th messages.

The problem parameters $\mu_1, \ldots, \mu_K$ are of course unknown to the IoT device, so to maximize its cumulated reward, it must learn the distributions of the channels, in order to be able to progressively focus on the best arm (i.e., the arm with largest mean). This requires to tackle the so-called exploration-exploitation dilemma: a player (here, an IoT device) has to try all arms (here, channels), a sufficient number of times to get a robust estimate of their qualities, while not selecting the worst arms too much.

We use here the UCB_1 algorithm which is known to be efficient for stationary *i.i.d.* rewards and is shown below.

3.2 The UCB_1 algorithm

A first naive approach could be to use the empirical mean estimator of the rewards for each of the K channels and select the channel with the highest estimated mean at each time. However, this "greedy" approach is known to fail dramatically [6]. Indeed, with this policy, the selection of arms is highly dependent on the first draws: if the first transmission in one channel fails and the first one on other channels succeeds, the

device will never use the first channel again, even it is the best one, which is the most available one in average.

Upper confidence bounds (UCB) algorithms instead use a confidence interval on the unknown mean μ_k of each arm, which can be viewed as adding a "bonus" exploration to the empirical mean. They follow the "optimism-in-face-of-uncertainty" principle: at each step, they play according to the best model, by selecting the statistically best possible arm (i.e., the highest upper confidence bound). More formally, for one IoT device, we denote by

$$T_k(t) = \sum_{\tau=1}^{t} \mathbb{1}(C(\tau) = k) \tag{2}$$

the number of times the channel k was selected up-to time $t \geq 1$. The empirical mean estimator of channel k is defined as the mean reward obtained by selecting it up to time t,

$$X_k(t) = (1/T_k(t)) \sum_{\tau=1}^{t} r_k(\tau) \mathbb{1}(C(\tau) = k) \tag{3}$$

For UCB_1 [7], the confidence term is

$$A_k(t) = \sqrt{\alpha \log(t)/T_k(t)}, \tag{4}$$

and the upper confidence bound is the sum of the confidence term and the empirical mean,

$$B_k(t) = X_k(t) + A_k(t), \tag{5}$$

which is used by the device to decide the channel for communicating at time step $t + 1$:

$$C(t+1) = \underset{1 \leq k \leq K}{\operatorname{argmax}} B_k(t) \tag{6}$$

The UCB_1 algorithm is called an index policy. It uses a parameter $\alpha > 0$, originally set to 2 [13], but empirically $\alpha = 1/2$ is known to work better (uniformly across problems), and $\alpha \geq 1/2$ is advised by the theory [13]. This algorithm is simple to implement and to use in practice, even on embedded microprocessors with limited computation and memory capabilities. In our model, every IoT device implements its own UCB_1 algorithm, independently. For one IoT device, the time t is the total number of sent messages from the beginning, as rewards are only obtained after a transmission. Different devices do not share this time index t as time is not slotted.

3.3 Multiplayer bandit issue

We can prove that one single intelligent IoT can improve consequently its performance in LPWAN IoT networks using unlicensed band [14]. But we have also shown that even if there are a lot of intelligent IoT devices and the model of other surrounding IoT devices does not stay purely stochastic, learning still brings improvement [9] [14]. Further theoretical developments on this direction are an interesting future work.

4 Measurement 1: IoT proof-of-concept

4.1 Preceding results

Bandit algorithms have been identified more than 10 years ago as efficient solutions for many cognitive radio problems, as introduced in [4]. In particular, the very trendy dynamic spectrum access (DSA [12]) issue has been identified as a multi-armed bandit (MAB) problem in [5]. The first implementation validating the bandit algorithms on real radio signals was presented 5 years ago for opportunistic spectrum access (OSA) in [8]. Reinforcement learning algorithms, such as UCB_1, were firstly used, but any kind of bandit algorithm [15] [16] [17] could be used indifferently. Their efficiency and implementation complexity can be considered criterion to decide which algorithm to implement. In the context of IoT, MALIN [18] is the first proof-of-concept (PoC) demonstrating the feasibility of using learning algorithms on the IoT device side, on real radio signals in lab conditions.

4.2 PoC setup

As illustrated in Fig. 2, this PoC is based on 4 USRP (Universal Software Radio Platforms) from Ettus Research and National Instrument[1] transmitting in the 434 MHz ISM bando (in order to not interfere with surrounding IoT networks usually deployed at 868 MHz). The development is made with GNU Radio[2] software, and the source code of the PoC is published on-line[3], in order to ease the full reproducibility of our results. We have not implemented a real IoT standard in this PoC, in order to show that it can be applicable for any IoT standard. However, global characteristics are rather corresponding to the LoRa context (not ultra-narrow band, reduced number of channels, frame duration around a few hundreds of milliseconds, etc.).

One or two (or more) USRP platforms are playing the role of IoT devices that can run (or not) the proposed learning algorithms, as platforms PF#3 and PF#4 of Fig. 2. So PF#3 and/or PH#4 (etc.) can play the role of one IoTligent device. They transmit at their own initiative some very light modulated information (using QPSK), in order to be identified by the gateway, and then wait during one second for the gateway ACK. Both uplink transmissions and downlink receptions use the same frequency channel. Whether the ACK is received

[1] See https://www.ettus.com/ for more details.
[2] See https://www.gnuradio.org/ for more details.
[3] See https://bitbucket.org/scee_ietr/malin-multi-armed-bandit-learning-for-iot-networks-with-grc. The code is released publicly under the open-source GPLv3 license.

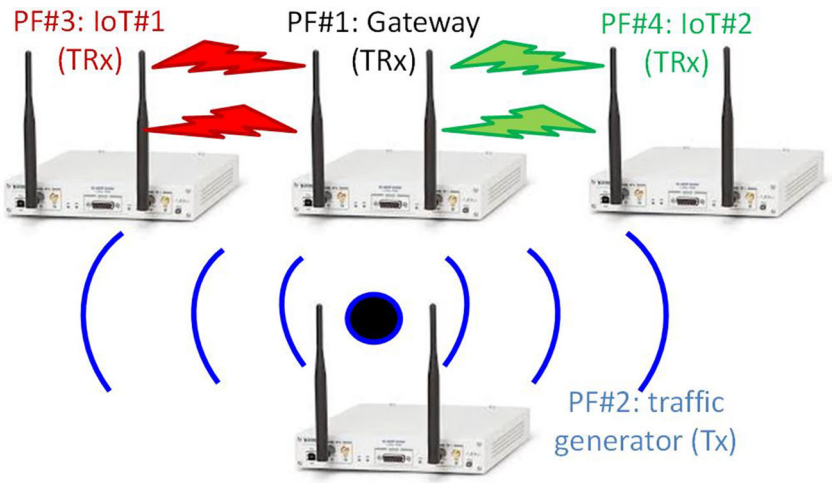

Fig. 2 PoC architecture composed of 4 USRP platforms programmed with GRC (GNU Radio Companion): PF#1, gateway; PF#2, traffic generator; PF#3, 4, etc., IoT devices (IoTligent devices if they run the learning algorithm or usual devices if they make a random access to the channels).

or not, the learning algorithm updates its knowledge about the channel used during this iteration.

USRP platform PF#2 is a traffic generator that emulates as much IoT traffic as we want, in order to be able to tune each channel's load independently. Traffic laws can be programmed independently on each channel on demand. For instance, we typically choose random i.i.d. channel loads ranging from 0 to 20%.

Last USRP PF#1 is a gateway (GW) that is continuously scanning all the K channels and monitors the IoT traffic composed of the artificial signals produced by the traffic generator and the IoT devices signals. The gateway has the ability to answer to the IoT devices, by sending back to them an ACK message, which contains their identifier (actually, the symbols corresponding to the QPSK complex conjugate of their identifier).

4.3 PoC results

The number of IoT channels K is a parameter, and we can set it to 4, 8, and 16 channels in our experiments, but there is no limitation. For the sake of clarity in the figures, we give examples below with 4 channels that are separated by empty channels, but they could be contiguous with no change neither in the implementation nor in the results. Only one IoTligent device is running in this experiment.

We can see on Fig. 3 a time-frequency waterfall view captured by the gateway, where we can observe the RF traffic in the $K = 4$ channels. The y-axis for the time is vertical and goes down, and frequency is on the x-axis. The difference of colors on the waterfall view is a difference of received power, due to the distance of the transmitters to the gateway receiver antenna during the experiment. The gateway transmitter antenna is very close so signals transmitted by the gateway are red. The traffic generator and IoT devices are a little bit further away, so the gateway received weaker signals from them: one is blue and the other green, which reveals a low difference.

In this experiment, we can see on Fig. 3 that channel #0 on the left hand side faces a dense IoT traffic, which appears as blue short transmissions (produced by the traffic generator). Some others uplink transmissions appear on channel #1 (second left hand side), but we do not see any blue short messages on channel #2 (third left) and #3 (on the right hand side, empty in this measure). However, we see on these channels longer messages of two kinds: green messages which correspond to IoT device transmissions, and red messages that are the answer done by the gateway. In order to rapidly have results in the demo, we make IoT device transmit roughly every 5 s, for a message duration of 1 s. Then when an IoT device transmits a message, the gateway should answer and send an ACK to the IoT device within 1 s, if the gateway was able to demodulate the signal, i.e., if there is no collision in the radio channel. For instance, we can see on Fig. 3 that the IoT device moved from channel #2 to channel #1, and at each of its transmission, the gateway was able to answer, by successfully sending an ACK response.

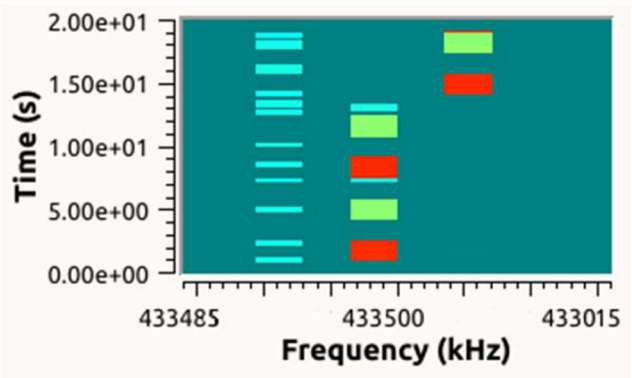

Fig. 3 Spectrum waterfall view on GRC received at gateway side in a 4 channels example (only 3 occupied in this picture), during experiments. Time is in y-axis (going down) and frequency in x-axis. Blue short transmissions are those produced by the traffic generator, green blocks are our IoT transmissions, and red blocks are the gateway transmissions itself

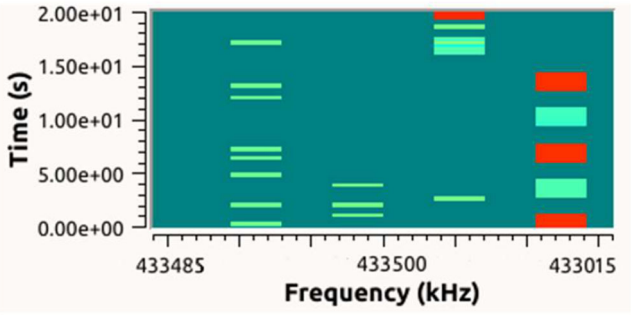

Fig. 4 Spectrum waterfall view on GRC received at IoTligent device side in a 4 channels example, during experiments. Time is in y-axis (going down) and frequency in x-axis. Green short transmissions are those produced by the traffic generator, red blocks are the IoT device transmissions, and blue blocks are gateway transmissions.

Fig. 4 gives the perspective of the IoTligent device, at a different moment for the same scenario. Then we observe that colors have changed, as the received power is now taken at the device side. The IoT device transmitter antenna is now very close, so signals transmitted by the IoT device are red. The traffic generator (representing other IoT devices' traffic) and then the gateway all are a little bit further away, so the IoT device received weaker signals from all of them, one is blue and the other green but inversed. However, it is not so obvious, so it is better to consider the message duration in the y-axis indeed.

We can see on Fig. 4 that if we use the same scenario of traffic as in Fig. 3, but at a different time, i.e., with a very dense traffic on channel #0, less dense on channel #1, even less dense on channel #2, then transmission appears on channel #3, but it is indeed just even less dense. At that time of the experiment, our IoTligent device is moving from channel #2, where maybe it faced some collisions in the downlink transmission of ACK, to channel #3, where several successive transmissions and receptions seem to occur.

Figure 5 is a screenshot taken at some moment during an experiment that gives the details of the learning algorithm operation embedded in IoTligent device. We can see in the top-left red data the number of selections of each channel. There is a clear disequilibrium with channel #3 that has been much more (17 times) used than channel #2 (8 times), itself more used than channel #1 (6 times) and channel #0 (only once). This reveals the effect of the learning algorithm. It has analyzed which channels are more occupied and more disturbed by other users of the band (emulated here by the traffic generator). The top-right green and therefore the bottom-right blue data explain such a choice. Channel #4 has known 16 successes (over 17), so a rate of 94%. We remind that successes mean that the IoT device received on that channel 16 ACK from the gateway after transmitting 17 times in this channel. So just one "exchange" was lost, either in UL or in DL, due to a collision with some interfering signal in the channel. We can see on the opposite that no success has been obtained for channel #0, so it has a 0% rate. UCB data, in bottom-left green part, are harder to follow, as UCB_1 indexes rapidly converge to very close values, but at each transmission, the IoT device chooses channel with highest UCB_1 index, as in (6).

5 Experimental architecture and hardware configuration for real LoRa measurements

The next step after the previously exposed proof-of-concept consists in implementing the same approach in real conditions of operation, that is, in a real IoT network and not only in laboratory conditions. We target here a LoRaWAN [1] [19] IoT context in the 868 MHz band, but it could be done with any other IoT standard, as soon as it uses ACK feedback. We describe the involved implementation details in this section.

As far as the authors know, this is the first implementation of decentralized artificial intelligence algorithms in IoT devices to tackle the IoT spectrum contention mitigation problem. It is first necessary to remind quickly how a LoRaWAN network is constituted. We are using here a real LoRa network in the European ISM band, at 868 MHz.

5.1 LoRaWAN architecture

The implementation of the learning algorithm we propose is decentralized, which means that it takes only place on the LoRa device side. As stated earlier, it impacts no aspect of the LoRaWAN network. We explain below a little bit more the LoRaWAN network side configuration, and we refer to [19] for more details. LoRaWAN network, as any other IoT network, can be summarized by four main elements, as shown in Fig. 6:

- LoRa IoT devices (IoTligent devices run the UCB_1 algorithm here)
- One or more LoRa gateway(s) receiving all LoRa radio signals in their radio range
- A LoRa network server (LNS) that discriminates devices subscribing to its network from others
- An Application Server (AS) that receives the data sent by devices and sends back ACK to the devices (mandatory here)

At their very first transmission, the IoT devices are associated to a given LoRaWAN network during a "join phase," through a gateway of this network. The LNS is

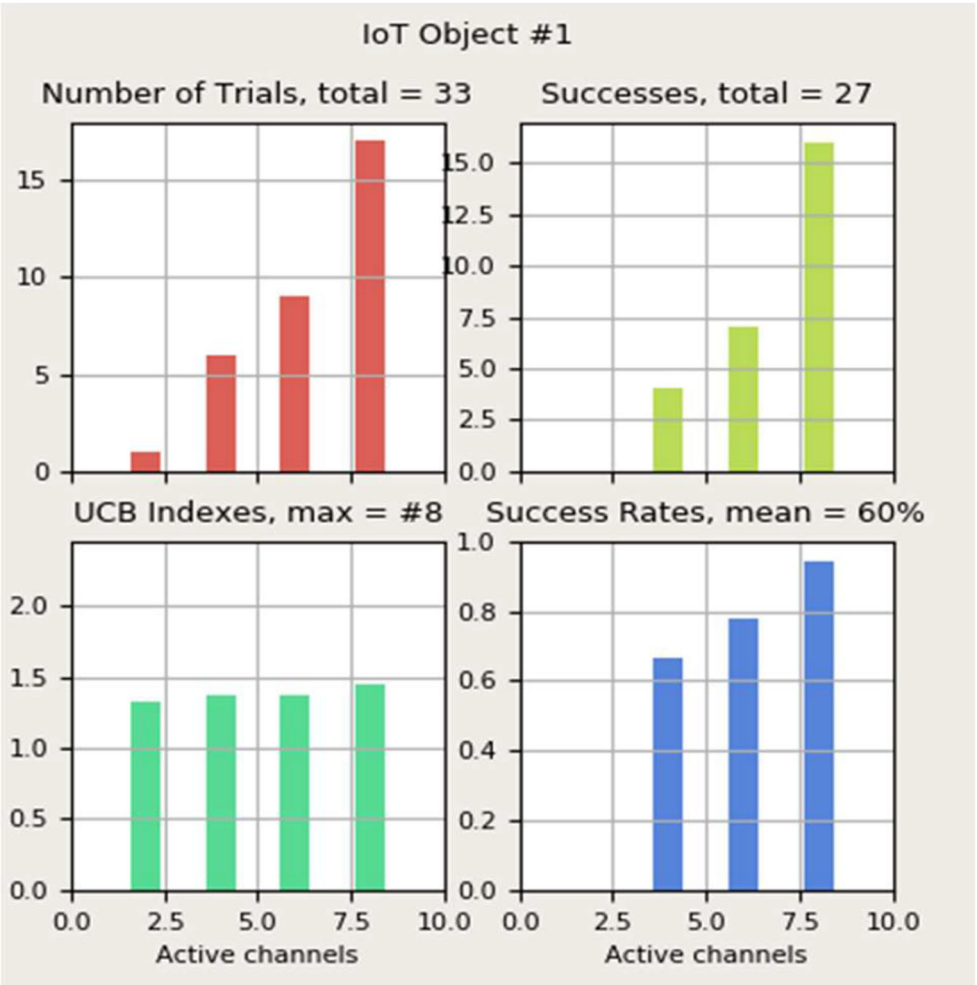

Fig. 5 Live results enabling to monitor the learning algorithm evolution at the IoTligent device side in a $K = 4$ channels example. Top-left red, number of trials on each channel; top-right green, number of successes on each channel (ACK received by IoT device); bottom-left green, UCB_1 index *Bk(t)* for each channel; and bottom-right blue, success rate on each channel

in charge of the association, as explained below. Finally, data extracted from radio signals, sent by the IoT devices, are sent to the application server (AS) that manages data (i.e., processes them, sends them to a storing place in the cloud and/or an application). Then the role of the AS is to launch the sending of an ACK to the IoT device, through LNS and a gateway, down to the IoT device.

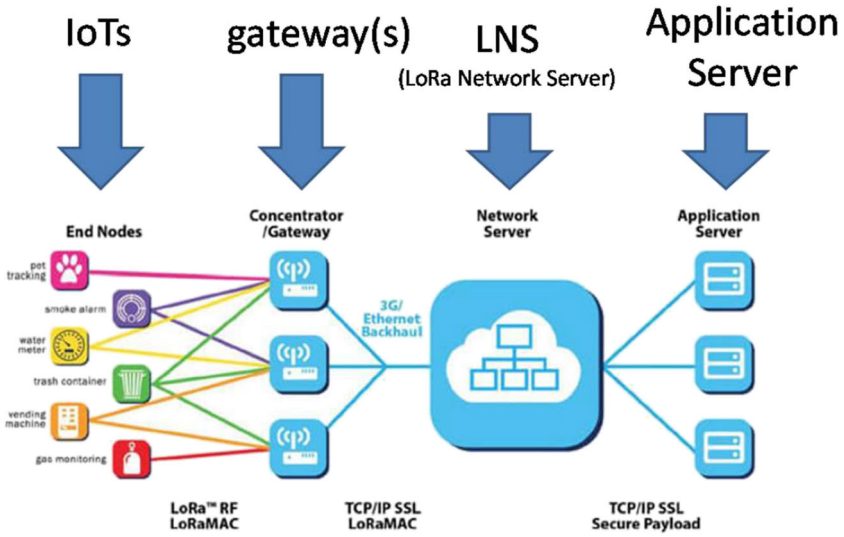

Fig. 6 LoRaWAN network parts: IoT devices, gateways, LNS, and AS [19]

5.2 Device side

For this experiment, we implement an IoT device by using a Pycom card[4] composed of an Expansion Board and a LoPy 4 module which can support LoRa wireless connectivity, as shown on Fig. 7. The Pycom card is programmed in the MicroPython language. The frequency channels used in the experiments are those authorized in France, the country of experimentation. The IoTligent proposal is agnostic to K, the number of channels in the standard, and thus, it can be used in any country.

We had to make several modifications in the LoRa library written in C and the ESP32 chip library written in MicroPython. By default indeed, the Pycom configuration for Europe is to use only 3 channels in a random access manner: 868.1, 868.3, and 868.5 MHz (with a duty cycle of 1%). So, for measurement 3 of Section 7 with more channels involved, we added a custom configuration region in the LoRa library with 16 channels, covering the band from 865.9 to 868.9 MHz. Moreover, we added the possibility for ESP32 chip to force a channel in *LORAWAN* mode, what is necessary in order to follow UCB_1 policy for both measurements 2 and 3 of the two following sections.

5.3 LoRa gateways

For measurement 2 of the next Section 6, we use outdoor LoRa gateways operated by Acklio Company that has several gateways deployed in the city of Rennes, where the experiments were made. As we did not have access to their configuration, only the 3 default channels have been used for this measurement campaign.

For measurement 3 of Section 7, we use our own indoor gateway shown in Fig. 8, whose channel parameters could be changed in order to adapt the number of channels depending on the measurement needs. This is done by changing the configuration file of the Semtech SX1301 chip which manages two radio SX1257 chips. It consists in choosing the central frequency of the two radio chips and choosing the offset in an interval of ± 500 k Hz, for each channel.

5.4 Network side

We have access to the LNS provided by Acklio Company. The LNS sends the received messages to an AS which is a Linux server, running in the cloud. The AS is running a Python program that enables to display data and metadata (i.e., frequency, time of reception, etc.). This programs also contains instructions to send an ACK to the device, using in DL the same frequency used by the IoT device at UL.

[4] Pycom documentation: https://GitHub.com/PyCom/PyCom-libraries

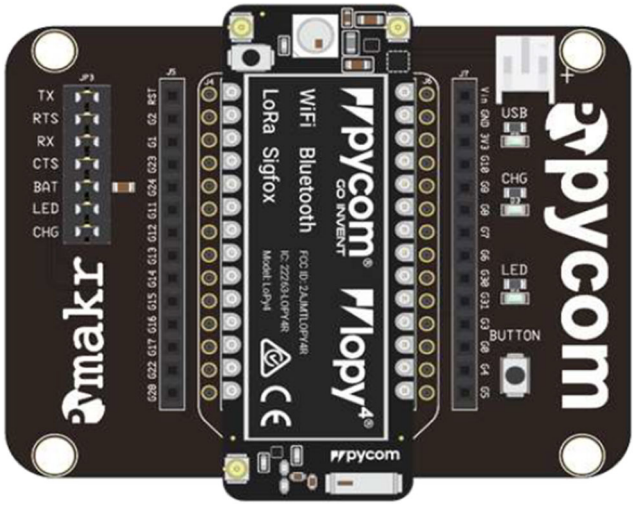

Fig. 7 Pycom module composed of a LoPy4 and an Expansion Board

6 Measurement 2: IoTligent operation in a real LoRaWAN network

6.1 Device side configuration

We use the *LORAWAN* mode at 868 MHz with an over-the-air-activation (OTAA) using *app_EUI* and *app_key* keys, as shown in the following MicroPython code for the Pycom device:

```
lora = LoRa(mode=LoRa.LORAWAN, region=LoRa.EU868)
lora.join(activation=LoRa.OTAA,    auth=(app_EUI,
app_key), timeout=0)
```

The transmit channel frequency is then chosen in a set of K channels, which is set here at $K = 3$ in this experiment. We use standard Europe UL channels with the following frequency table (in Hz):

```
tabFreq =[868100000, 868300000, 868500000]
```

The IoTligent device infinite *while loop* is started, running the algorithm presented in the Section 3.2 and [5], in order to choose which frequency to be selected at each iteration before

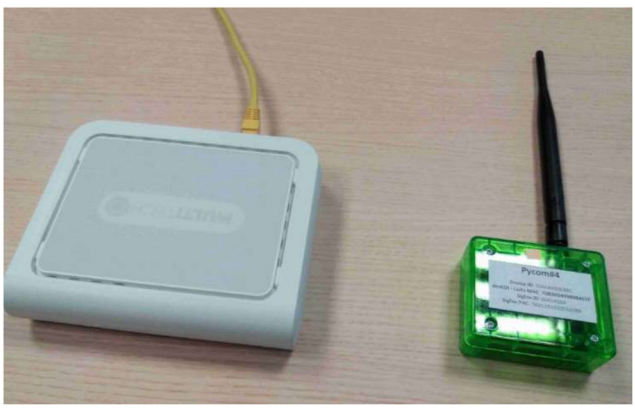

Fig. 8 Indoor gateway used for measurement 3 experiments on the left side and packaged Pycom device on the right side

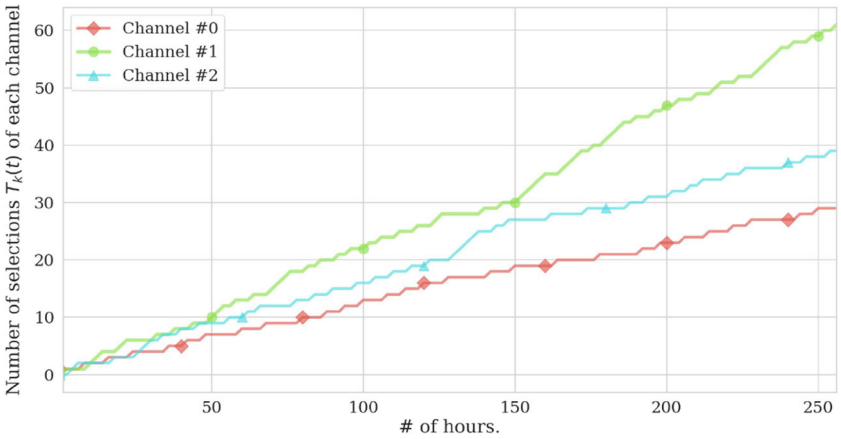

Fig. 9 Evolution of the T_k index through time, as learning happens

executing a send operation. An ACK is then expected from the network side in *non-blocking* mode so that when ACK is not received, the device just updates its learning data and still goes on.

6.2 Network side – LoRa network server (LNS)

The different IoTligent devices should be declared to the LNS, with at least the following information:

- *devEUI*: ID of the device obtained by executing a « *get_id.py* » program[4] on the Pycom device itself.
- *appEUI*: which should correspond to *app_eui* chosen in the Pycom device.
- *appKey*: which should correspond to *app_key* chosen in the Pycom device.
- Other parameters are let by default at SF = 12 (spreading factor) and bandwidth BW = 125 kHz.

The address of the AS is also specified in *Connectors*, as well as the mode used to send data between LNS and AS (we chose *http callback* here).

6.3 Network side – application server (AS)

The AS runs a Python program that receives data from the LNS, as well as LoRa metadata with all parameters of the LoRaWAN transmission (frequency, SF, BW, time of arrival, etc.). This program also sends an ACK message to the device in DL. First, an acknowledgement attempt is sent by default at the same frequency than the message transmitted by the device it answers to. Then we block any other retransmission. This is exactly what is necessary for the learning process of IoTligent:

- To use the same channel in both UL and DL
- To avoid retransmission in order to increase the battery durations of devices on the one hand and radio frequency overload on the other hand.

6.4 Learning algorithm in Pycom device

The learning algorithms used in IoTligent are (any) bandit algorithms, such as those first used for cognitive radio dynamic spectrum access in [5], and implemented in the exhaustive open-source SMPyBandits Python library [17].

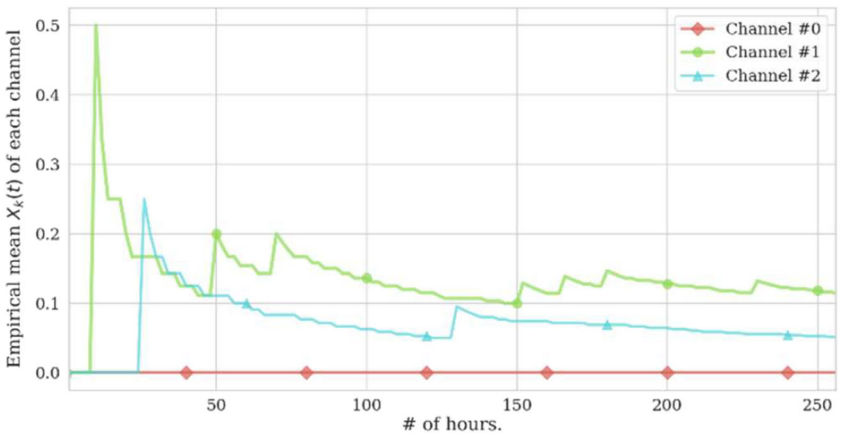

Fig. 10 Evolution of the X_k empirical mean through time

Table 1 Results at the end of the experiment

Channel #0	Channel #1	Channel #2
$T_k[0] = 29$	$T_k[1] = 61$	$T_k[2] = 39$
$X_k[0] = 0.0$	$X_k[1] = 0.115$	$X_k[2] = 0.051$
$S_k[0] = 0$	$S_k[1] = 7$	$S_k[2] = 2$

We take here the example of UCB_1 algorithm, as presented above. We have chosen this algorithm as it is known to be efficient and to converge quickly and also for its ease of implementation. The only data necessary to be stored for the UCB_1 algorithm are:

- An iteration index initialized at 0: `it`
- A table of size N (the number of channels, 3 in this implementation example, but it could be arbitrarily high) for the number of times each channel has been chosen, representing T_k of (2): `Tk[ ]`
- Another table of size N for the empirical mean of success of each channel, i.e., $X_k(t)$ of (3): `Xk[ ]`

From the point of view of the learning algorithm, a success occurs when an IoTligent device receives an ACK from the IoT network (as explained above), which means that the currently used frequency channel suffered no collision in both UL and DL. Otherwise, a failure occurred. The update of the selected channel empirical means X_k is reconstructed easily from the number of activations and the previously stored X_k value. Therefore, it is not necessary to store in memory the results of all past iterations, but just only a summary of it (its mean). The proposed solution is thus realistic and efficient, as it only requires a bounded storage capacity.

Then, after an initialization phase where each channel is selected alternatively, once the channel selection really starts to use the UCB_1 indexes [5]. It consists for each iteration in choosing the frequency channel with the greatest index B_k as defined in (5), that is, computed for each channel like this in a *for loop* on *i* index, and with *alpha* the UCB_1 parameter α that controls the exploration vs. exploitation trade-off [5]:

$$\mathrm{Ak[i]} = \mathrm{sqrt(alpha*log(it)/Tk[i])}$$

The IoTligent device then selects as in (6) the channel having the greatest UCB_1 index B_k [5]:

```
for i in range(N) :
  Bk[i] = Xk[i] + Ak[i]
   if Bk[i] > bestChannel :

        bestChannel = Bk[i]; freq = tabFreq[i]
```

6.5 Results for the second measurement

Experiments have been done on a real LoRa network currently deployed with $K = 3$ channels. We present results obtained on an IoTligent device, for 129 transmissions done every 2 h, for a period of 11 days. Figure 9 shows the evolution of the T_k index through time, which is the number of time each channel has been selected by the learning algorithm. In the figures, the red curve is for channel #0 (at 868.1 MHz), the green curve is for channel #1 (868.3 MHz), and the light blue curve is for channel #2 (868.5 MHz).

Figure 10 gives the empirical mean *Xk* experienced by the device on each of the 3 channels. Each peak corresponds to a successful LoRa bi-directional exchange between the device and the AS: from the device UL transmission to the ACK reception (DL) by the device.

We can see that channel #1 gives the best results, before channel #2, but channel #0 always failed in sending back an ACK to the device. Each peak in Fig. 10 reveals a successful case where an ACK has been received by IoTligent device. Table 1 gives the end results after 11 days. We can see that channel #0 has been tried 29 times with $S_k[0] = 0$ success (i.e., no ACK received by the device). So the learning algorithm made the device use 61 times channel #1 with $S_k[1] = 7$ successful bi-directional exchanges and 39 times channel #2 with $S_k[2] = 2$ successes. This corresponds to 7 (respectively 2) peaks of $X_k[1]$ (respectively $X_k[2]$) on Fig. 10.

The empirical mean X_k gives the vision the device obtained from the channels, i.e., a mean probability of 11.5% of successful bi-directional connection for channel #1 and 5% for channel #2, whereas channel #0 never worked from the device point of view. With a normal device, i.e., a non-IoTligent device, that uses a purely random access, trying once over 3 times on each channel, for a global average successful rate of 5.5%.

It is important to note that here the learning algorithm is mostly in its exploration phase, but it is learning very fast. Only during the last 2 days of the experiment, channel #1 has already been used 4 times more than channel #0 and 2.5 times more than channel #2, which means that learning is already very effective. As proven for UCB algorithms [6, 8], channel 1 will be more and more selected so that the global success rate will converge to the percentage of success of the best channel, which is 11.5% in this experiment (this estimate can be considered a good evaluation as it is based on 61 trials). In other words, a mean of 15 successes can be expected in the long term over the same period of 11 days with IoTligent. On the contrary, normal devices will never improve and stay in the current average, i.e., in average 7 successful transmissions on the same period duration.

In order to have the same rate of successful transmissions, normal IoT devices should consequently transmit twice *more* often, which has two negative impacts. The first impact is that

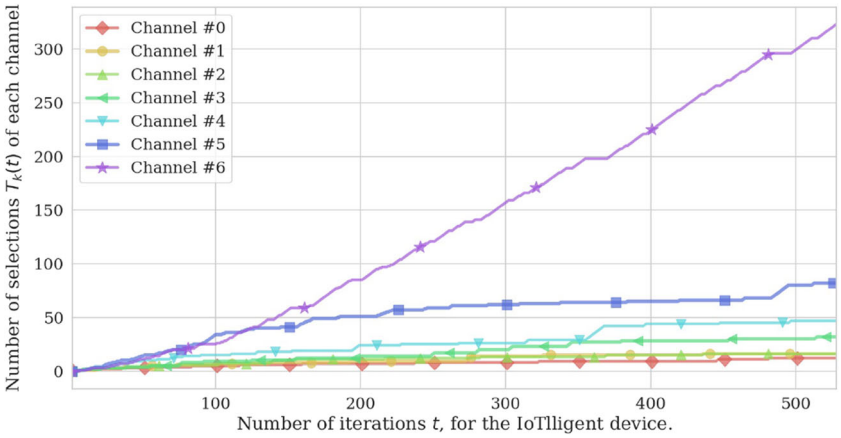

Fig. 11 Evolution of the number of selections through time of IoTligent device for scenario 1

normal IoT devices autonomy will be twice *less* than IoTligent devices. The second but not the least impact is that devices will occupy twice more times the radio channels, hence contributing to increase even more the risks of radio collisions and thus the IoT bands congestion.

7 Measurement 3: IoTligent operation in a LoRaWAN network with emulated artificial traffic

As a way to make a complete validation of IoTligent proposal, we now combine the two previous experiments by running IoTligent real LoRa IoT devices on a real LoRaWAN network at 868 MHz, but under the future expected heavy IoT networks load, emulated using USRP platforms.

7.1 Experimental setup

As far as we know, this is the first evaluation in a real LoRaWAN network of LoRa devices running on-line learning algorithms, with emulated traffic reproducing very dense IoT conditions. The measures use a Faraday cage and an anechoic chamber, in order to avoid jamming real LoRaWAN networks operating in the surroundings of the laboratory. It also enables to be fully in control of the ISM jammers as well as the channel propagation, and to perfectly monitor what is happening during the measurement campaigns. As for the first PoC measurement, we use one (or several) USRP platform as a traffic generator, in order to emulate the ambient traffic made by the surrounding IoT devices. Each channel's occupancy rate can be set independently on demand, so that it is non-uniform over the channels. The experiments presented below used a set of $K = 7$ channels, with different colors in the next plots:

- Channel #0: 866.9 MHz, in red
- Channel #1: 867.1 MHz, in orange
- Channel #2: 867.3 MHz, in light green
- Channel #3: 867.5 MHz, in green
- Channel #4: 867.7 MHz, in light blue
- Channel #5: 867.9 MHz, in dark blue
- Channel #6: 868.1 MHz, in purple

For each experiment, we compare the results of two LoRa IoT devices: (i) one *IoTligent device* running the learning algorithm (UCB_1) and (ii) one usual LoRa device that acts as a

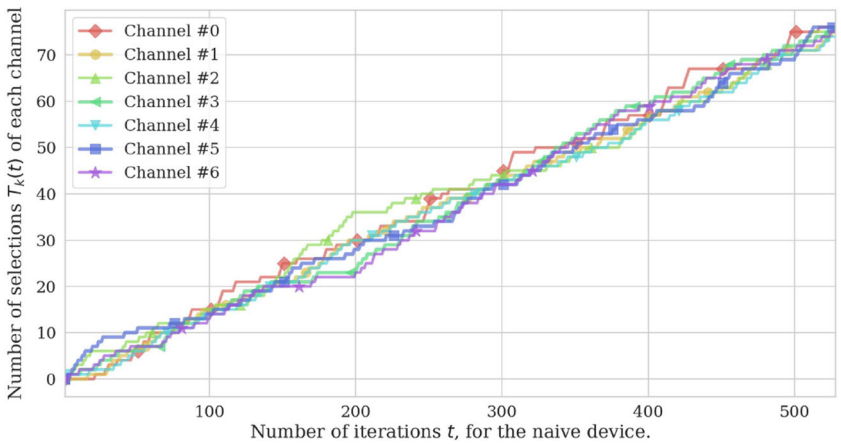

Fig. 12 Evolution of the number of selection through time, for the reference (naive) IoT device for scenario 1

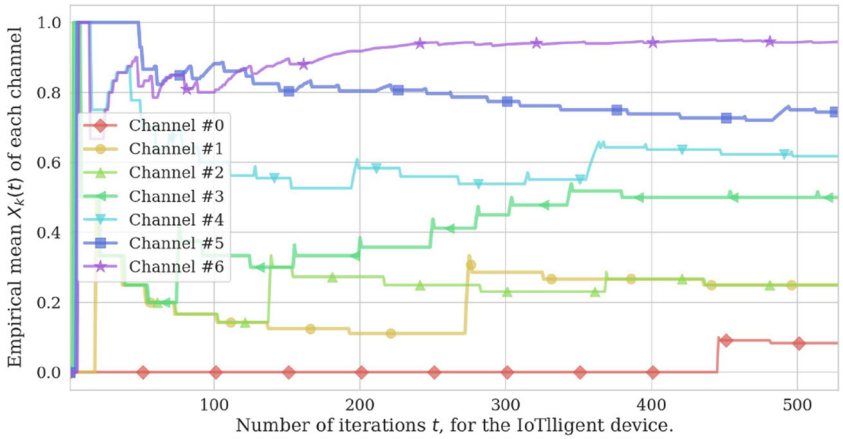

Fig. 13 Evolution of the X_k empirical mean through time of IoTligent device for scenario 1

reference and that we name *reference IoT device*. The gain obtained by the proposed approach can be directly measured by the difference between the number of successful communications obtained by IoTligent device compared with the results of the reference IoT device, as both run in the same conditions of traffic load. It can also be made by a comparison of their success rate.

During the experiments, we make devices transmit every 20 s. A successful communication occurs when the IoT device receives in DL an ACK to its own last UL transmission (on the same channel). In that case, the gateway receives messages on the frequency channel selected by the devices, i.e., randomly for the reference device, or by running the bandit algorithm for the IoTligent device. The gateway then forwards the message to the AS through the LNS. The acknowledgement is then sent back to the opposite way and the gateway uses the same channel as the one used by the device in uplink, regardless if the device is IoTligent or not. As only these two devices are requesting acknowledgements and no other real LoRa device can access the gateway in the chamber, the constraint of 1% duty cycle is not exceeded by the gateway in any channel.

We ran the experiments over several hundreds of iterations (i.e., of transmissions) so that they have a duration of a couple of hours. We used USRP platforms as jammers that generate emulated IoT traffic. As the USRP transmission power, for jamming signal, is not as high as those of LoRa IoT devices and LoRa gateway, LoRa IoT devices power has been decreased with attenuators of 40 dB. However, this has not been possible to do it on the gateway, and we discuss the consequence below.

For one device, when no acknowledgement is received, it means that there has been a collision either in DL or in UL. Here we can assert that collisions only occur in UL as we can check that at each time a message has been received by the AS, an ACK has been correctly received by the IoT device (reference or IoTligent). This is because the gateway DL feedback power is very high compared with the USRP jamming level.

We now detail here a couple of scenarios that have been executed for measurement 3, one with a medium density context of IoT devices and second one with even more dense conditions.

7.2 Scenario 1: Not too heavy traffic and one free channel

We choose in scenario 1 a context where channels occupancy is slightly decreasing from one to another. This enables to understand how the algorithm runs as a first approach. The

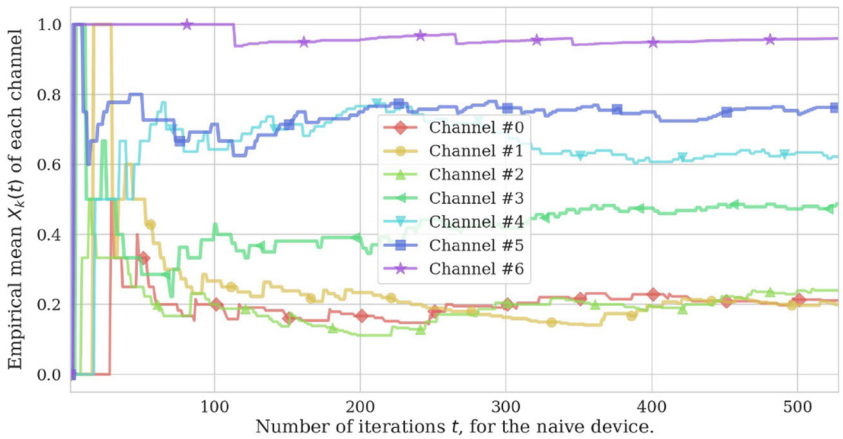

Fig. 14 Evolution of the X_k empirical mean through time of reference IoT device for scenario 1

Table 2 Compared results at the end of the experiment between reference IoT device and IoTligent device in terms of number of activations and percentage of success on each channel for scenario 1

	Reference IoT		IoTligent	
Channel	% of success	Nb of activations	% of success	Nb of activations
#0	21 %	76	8 %	12
#1	20 %	76	25 %	16
#2	24 %	75	25 %	16
#3	49 %	76	50 %	32
#4	62 %	74	61,7 %	47
#5	76,3 %	76	74,4 %	82
#6	96 %	75	94,4 %	**323**

percentage of occupancy for each channel is 30% for channel #0, 25% for channel #1, 20% for channel #2, and so on decreasing of 5 % at each step until 0% for channel #6, so a vector like this for the $K = 7$ channels: $\{0.30, 0.25, 0.20, 0.15, 0.10, 0.05, 0\}$. So the channel where less radio collisions should occur is obviously channel #6 and we name it the *best channel*.

We visualize in Fig. 11 below the number of times each of the channels has been used by IoTligent device through time. The purple curve of channel #6 is clearly leading the race, followed far away by dark blue channel #5, then light blue channel #4, and so on. It allows to conclude that the learning algorithm has clearly been able to favor the use of the less occupied channels by the LoRa IoTligent device. Moreover, after a short time, it has been able to make a difference between each different level of occupancy in the channels, with a great advantage to the best one.

To give a rigorous comparison, we can see on Fig. 12 that the reference LoRa device, on its side, has uniformly used the 7 channels during the experiment. This perfectly illustrates the difference between an IoTligent and a usual (naïve) IoT device. IoTligent device uses reinforcement learning to make the choice of a channel before each transmission, based on a given metric (reward) depending on its past experience. Here the followed policy is UCB_1 (with exploration factor $\alpha = 2$).

The curves of Fig. 13 represent the empirical mean reward of (3) for each channel obtained by IoTligent device. With no surprise, the channels that have been the most used for transmissions are those with the best mean reward, i.e., the best percentage of vacancy. The same order is found as in Fig. 11, with purple curve of channel #6 first, followed by dark blue channel #5, then light blue channel #4, and so on. Indeed, Fig. 11 is a consequence of Fig. 13, as UCB_1 chooses more and more with time, as stated in (5), the channels with higher empirical means X_k, as the A_k term decreases with the number of activation T_k and becomes negligible compared with X_k. Note that jumps in the curves are having bigger steps on the channel curves that have been used less often, as A_k of (5) is still predominant when T_k is low.

We also monitor the reference device empirical mean reward in Fig. 14, but just for comparison purposes as it is not used by the reference IoT device to select its channels. The difference with the IoTligent device is that reference device has played more times the bad channels and less times the best channels, as expected. As a consequence, there is a little bit less variance on the empirical mean reward for less occupied channels for the reference IoT than for the IoTligent device but the opposite for most occupied channels. However, from IoTligent point of view, the goal is not to recover the empirical mean, but to order the channels in terms of their occupancy rate.

In Table 2. the number of activations and percentages of successful transmission obtained on each channel by the IoTligent and reference IoT devices are listed. Whereas the reference IoT device uniformly transmits on all channels, we can see that the IoTligent device has been concentrating on most vacant channels, with a clear choice for the less occupied channel #6. Over 526 iterations, 323 transmission have been done in this channel, i.e., more than 4 times compared with the reference IoT (with 75) and 27 times more than for channel #0 for IoTligent device (with 12). This gives the opportunity to the IoTligent device to increase drastically its global successful rate that can be seen on Table 3. The IoTligent solution allows the device to reach a successful transmission rate of almost 80% against 50% for the reference IoT device.

Table 3 emphasizes the advantage of IoTligent solution as it almost improves by 2.5 times the performance of reference device, in terms number of failure (when ACK is not received by the device). This is due to the ability of IoTligent to favor

Table 3 Compared results at the end of the experiment between reference IoT device and IoTligent device in terms of percentage of success (i.e., ACK received by the device) for scenario 1

	Reference IoT	IoTligent
Nb of iterations	528	528
Nb of no ACK	266	108
% of success	49.6 %	79.5 %

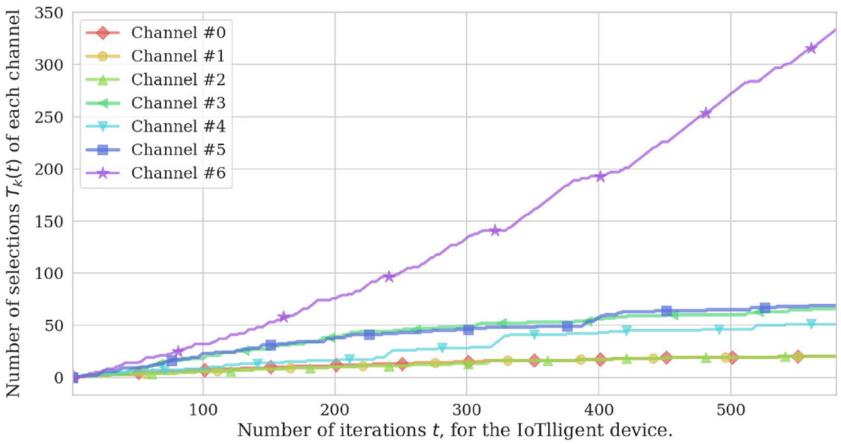

Fig. 15 Evolution of the *number of selection* through time of IoTligent device for scenario 2

the use of less occupied channels and especially channel #6 which is completely vacant.

7.3 Scenario 2: Very heavy traffic

Let us consider now a heavy traffic scenario as percentage of occupancy for each channel is set to 40% for channel #0, #1, and #2, 30% for channel #3, 20% for channel #4, 15% for channel #5, and 10% for channel #6. So once again, channel #6 is the *best channel* but far from being unoccupied this time.

As in the previous scenario, we illustrate on Fig. 15 that channel #6 is the most played one, even if it is not fully vacant. Indeed, the learning algorithm takes into account the relative occupancy rate between the channels. So the differences with the three following channels (#5, #4, #3) look like scenario 1, but clearly all the three first channels (#0, #1, and #2) are almost always avoided.

We see the empirical mean measured by the IoTligent device, in Fig. 16. It reflects once again the (inverse) order of the occupancy rate that has been set for the channels.

Figure 17 confirms the results in terms of mean success of channels by the reference IoT device, and we find approximately the same channel mean occupancy rates as for IoTligent case of Fig. 16. Results are even more solid for the reference device as all channels have been selected the same number of times. Here also, the IoTligent device has only gathered a few samples on the worst channels so that the evaluated empirical mean is not so realistic.

Table 4 shows the number of times each channel has been selected and the obtained empirical percentage of successful transmission. For both devices, we can see a direct (inverse) correspondence with the occupancy rate of each channel given earlier. Most importantly, we observe an unbalanced number of activations of the channels thanks to the learning algorithm. Whereas the reference device roughly uses each channel equally (around 80 times), we can see that the IoTligent device concentrates its transmission in channel #6 (334 times) as it provides the best percentage of success and neglects the worse channels (20 times).

We can also see on Table 4. that despite the IoTligent device experimented worse results than the reference IoT device on the best channel (channel #6), its global results are much better.

Results of Table 5 in terms of percentage of success show how efficient the proposed solution is. With a global mean of 28% of channel occupancy on the 7 channels, the reference IoT device obtains in this experiment a 32 % of transmission successes, whereas IoTligent device is able to reach 51%.

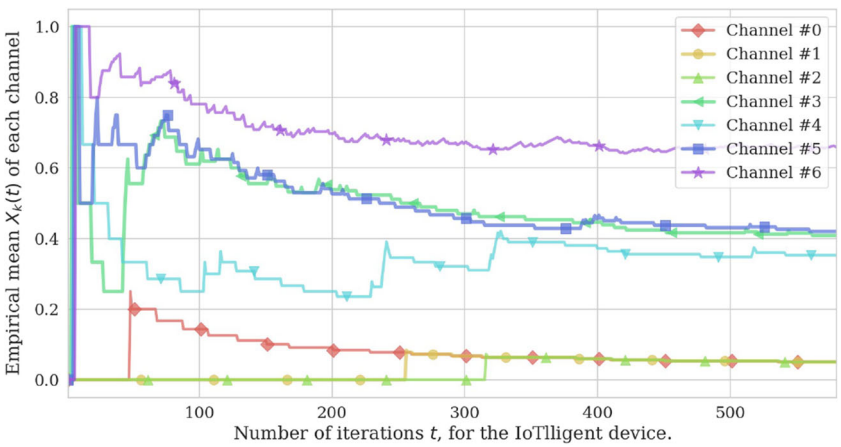

Fig. 16 Evolution of the X_k empirical mean through time of IoTligent device for scenario 2

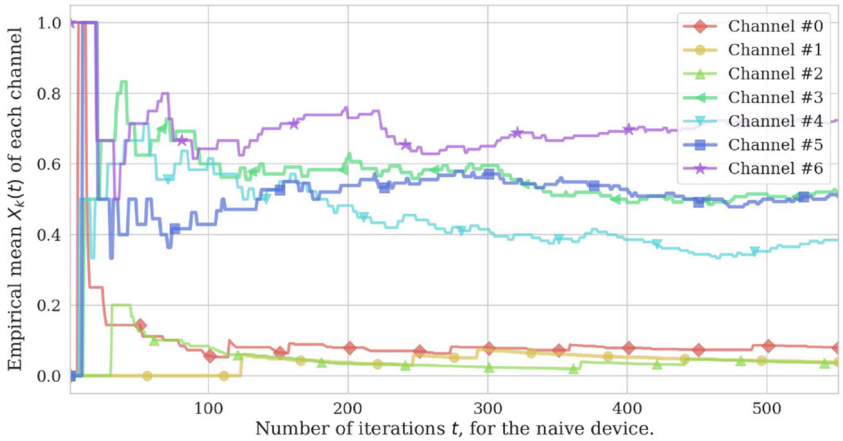

Fig. 17 Evolution of the X_k empirical mean through time of reference IoT device for scenario 2

Table 4 Compared results at the end of the experiment between reference IoT device and IoTligent device in terms of number of activations and percentage of success on each channel for scenario 2

	Reference IoT		IoTligent	
Channel	% of success	Nb of activations	% of success	Nb of activations
#0	7,9 %	76	5 %	20
#1	3,9 %	78	5 %	20
#2	3,5 %	85	5 %	20
#3	52 %	77	41 %	66
#4	38,5 %	78	35 %	51
#5	50,6 %	81	42 %	69
#6	72,4 %	76	**65,9 %**	**334**

Remark that these figures take into account the very beginning of the learning phase where the bandit algorithm is still exploring. Table 6 gives the results during the 100 last iterations for best channel #6. Percentage of success reaches 66.7 % for IoTligent indeed, which is coherent with the results of Table 4 where we can see that channel #6 percentage of success is almost 66 %. As IoTligent is now almost only targeting channel #6 (81 times over 100), its percentage of success is slowly sliding towards the empirical mean of this channel. This is exactly what is proven to be achieved at infinity by the mathematical proof of converge of UCB_1 [5]. However, we demonstrated that efficiency is obtained much earlier, in practical radio conditions.

7.4 Synthesis and extension to multiusers

This final measurement campaign definitely confirms that the proposed approach can be a solution for radio collision mitigation in the future IoT ultra-dense networks in the unlicensed bands. Our study in [9] confirmed by simulation that advantage still remains even if the number of IoTligent devices increases, using solution from the literature in order to orthogonalize IoTligent devices without coordination.

8 Conclusion

We describe in this paper the decentralized solution we propose to mitigate radio collisions in IoT unlicensed bands. Our solution is to use machine learning algorithms, to be implemented on the IoT device side, at a very low cost of implementation and no protocol overhead. We propose to use multi-armed bandit algorithms, and we first prove the efficiency of the method on a proof-of-concept demonstration based on USRP platforms in laboratory conditions. We then present the implementation of these MAB learning algorithms on devices deployed in a real IoT network, and finally we show the validity in the expected future conditions of massive IoT deployment. Implementation on LoRa devices in a real LoRaWAN network is demonstrated and is named

Table 5 Compared results at the end of the experiment between reference IoT device and IoTligent device in terms of percentage of success (i.e., ACK received by the device) for scenario 2

	Reference IoT	IoTligent
Nb of iterations	551	580
Nb of no ACK	373	283
% of success	32,3 %	51,2 %

Table 6 Compared results the last 100 iterations of the experiment between reference IoT device and IoTligent device in terms of number of activations and percentage of success for channel #6 for scenario 2

	Reference IoT		IoTligent	
Channel	% of success	Nb of activations	% of success	Nb of activations
#6	80 %	15	66,7 %	81

IoTligent. As far as we know, we propose the first implementation of a decentralized spectrum learning scheme for IoT wireless networks. Even if the current IoT networks are not (yet) densely populated by devices, medium and even short-term forecasts predict a high number of devices to overcrowd the ISM unlicensed bands. The IoTligent approach is then a solution to extend on the one hand the IoT devices battery life, which is a key performance indicator in any IoT ecosystem, and on the other hand to mitigate the collision issue that will occur with the growing number of IoT devices.

Acknowledgment The authors would like to thank Rémi Bonnefoi for the MALIN implementation as well Yalla Diop for their technical support on LoRa network and Pycom programming.

Funding This publication is supported by the European Union through the European Regional Development Fund (ERDF) and by the French Region of Brittany, Ministry of Higher Education and Research, Rennes Métropole and Conseil Départemental 35, through the CPER Project SOPHIE/STIC & Ondes.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Sornin N, Luis M, Eirich T and Beylot AL (2015) "LoRaWAN specification", technical report, LoRa Alliance, Inc.
2. Fourtet C (2015) "The technical challenge of future IoT networks and their consequences on modem's and SoC's design", Réseaux et services conference. R&S, Paris
3. Moy C (2019) IoTligent: first world-wide Implementation of decentralized spectrum learning for IoT wireless networks. URSI AP-RASC, New Delhi, pp 9–14
4. Jouini W, Ernst D, Moy C, Palicot J (2009) Multi-armed bandit based policies for cognitive radio's decision making issues. Signal Circuits and Systems Conference, Jerba, pp 6–8
5. Jouini W, Ernst D, Moy C and Palicot J, (2010)"Upper confidence bound based decision making strategies and dynamic spectrum access," *IEEE ICC, International Conference on Communications,* Cape Town, South Africa.
6. Lai TL, Robbins H (1985) Asymptotically efficient adaptive allocation rules. Adv Appl Math 6(1):4–22
7. Auer P, Cesa-Bianchi N, Fischer P (2002) Finite-time analysis of the multiarmed bandit problem. Mach Learn 47:2–3
8. Moy C (2014) "Reinforcement learning real experiments for opportunistic spectrum access", *Karlsruhe Workshop on Software Radio*. Karlsruhe, Germany
9. Bonnefoi R, Besson L, Moy C, Kaufman E, Palicot J (2017) Multi-armed bandit learning in IoT networks: learning helps even in non-stationary settings. CrownCom, Lisbon
10. Anandkumar A, Michael N, Tang AK, and Swami A, (2011) "Distributed algorithms for learning and cognitive medium access with logarithmic regret", IEEE J Selected Areas Commun 29(4)
11. Robbins H (1952) Some aspects of the sequential design of experiments. Bull Am Math Soc 58(5):527–535
12. Zhao Q, Sadler B, "A survey of dynamic spectrum access", in IEEE Signal Processing and Magazine, 2007.
13. Bubeck S, Cesa-Bianchi N (2012) Regret analysis of stochastic and non-stochastic multi-armed bandit problems. Found Trends® Mach Learn 5(1):1–122
14. Besson L (2019), "Multi-players algorithms for Internet of Things networks", PhD thesis, CentraleSupélec.
15. Thompson WR, (1933)"On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," Biometrika 5
16. Moy C, Palicot J, and Darak SJ, (2016) "Proof-of-concept system for opportunistic spectrum access in multi-user decentralized networks", EAI Endorsed Transactions on Cognitive Communications 2.
17. Besson L, "SMPyBandits: an open-source research framework for single and multi-players multi-arms bandits (MAB) algorithms in Python". Code on https://GitHub.com/SMPyBandits/SMPyBandits and documentation on https://SMPyBandits.GitHub.io/. Accessed: 2020
18. Besson L, Bonnefoi R, Moy C (2018) *MALIN: multi-armed bandit learning for Iot networks with GRC*: a TestBed implementation and demonstration that Learning Helps. ICT, France
19. LoRaWAN™, (2017) v1.1 Specification, LoRa Alliance Inc, https://LoRa-alliance.org/sites/default/files/2018-04/lorawantm_specification_-v1.1.pdf. Accessed: 2020
20. Moy C, Besson L (2019) Decentralized spectrum learning for IoT wireless networks collision mitigation. First International Workshop on Intelligent Systems for the Internet of Things, Santorini Island, pp 29–31

This journal paper is an extension of conference paper [20].

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.