GNU Radio implementation for Multiuser Multi-Armed Bandit learning algorithms in IoT networks

Julio Manco-Vasquez[1], and Christophe Moy[2], Faouzi Bader[1]

[1] IETR / CentraleSupélec Campus de Rennes, F-35510 Cesson-Sévigné, France,
{JulioCesar.MancoVasquez, Faouzi.Bader} @CentraleSupelec.fr
[2] Univ Rennes, CNRS, IETR - UMR 6164, F-35000, Rennes, France
Christophe.Moy@Univ-Rennes1.fr

Abstract

Novel access schemes based on multi-armed bandit (MAB) learning approaches has been proposed to support the increasing number of devices in IoT networks. In the present work, a GNU radio framework is implemented to recreate an IoT network where IoT devices embedding MAB algorithms are able to learn the availability of the channel for their packet transmissions to the gateway. It allows to incorporate several IoT users recognized by an identifier (ID), and provides a gateway to handle a large number of IDs as well as the packet collisions among IoT devices. The experimental results show that the introduction of learning mechanism in access schemes can improve the performance of the network.

1 Introduction

Several efforts to introduce reinforcement learning algorithms tailored for low-power wide-area (LPWA) networks have been recently carried out [1, and references therein]. However, unlike opportunistic spectrum access (OSA) schemes, where several proof of concept based on MAB algorithms have been developed [2, and references therein], the experimental evaluation for IoT networks has been overlooked.

Previous works regarding the evaluation of MAB algorithms for OSA in decentralized networks do not take into account realistic transmissions between the primary and secondary users, and utilize particular toolboxes avoiding to run reproducible experiments. In this regard, a first proof of concept to assess the potential usage of MAB algorithms for IoT scenarios is provided in [1]. It is fully implemented in GNU radio, and we consider this initial effort to introduce features concerning an LPWA network. In doing so, the emulation of an IoT network to support a large number of users is addressed. Our testbed is composed of several IoT devices and a Gateway, where each IoT device following an ALOHA wireless protocol transmits a packet containing its ID, and waits for an ACK packet transmitted by the gateway, as it is shown in Fig. 1. For that end, a data packet structure is implemented to provide the required support for a multiuser scenario. Our demonstration shows that significant gains can be obtained, when a well-known MAB approach, an Upper-Confidence Bound (UCB) algorithm [1] is embedded in IoT devices.

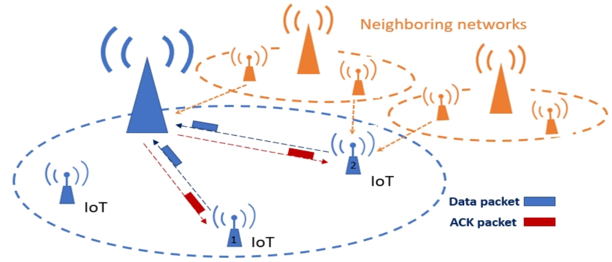


Figure 1: A LPWA scenario with IoT devices that aim to select the best channel for their transmissions to the gateway, while collisions among other IoT devices and interference from other networks may occur.

2 Experimental setup

Our testbed is implemented using N210 USRPs connected to an Octoclock for time and frequency synchronization among the devices. Each USRP running a GNU radio application implements a transceiver composed of three GNU radio blocks. For instance, in an IoT user, a first block corresponding to the physical layer detects and demodulates the packets into QPSK symbols, after which a second block detects the ID within the ACK packet. In the last block, a new packet is created and transmitted through the frequency channel pointed out by the MAB algorithm, if it is embedded in the IoT device [1].

The implemented data packet structure is shown in Fig. 2, where a preamble is utilized for the packet detection and phase correction of the received signal, while the values in the field UP/DOWN allow the receivers at the gateway and the IoT user to only receive uplink and

[1] In a similar way, it operates at the gateway side, where after demodulating the packet, an ID detection is carried out to identify the IoT user, and finally an ACK packet is created in the last block.

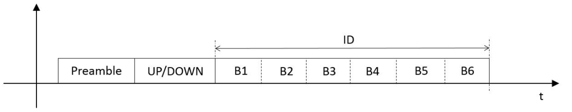
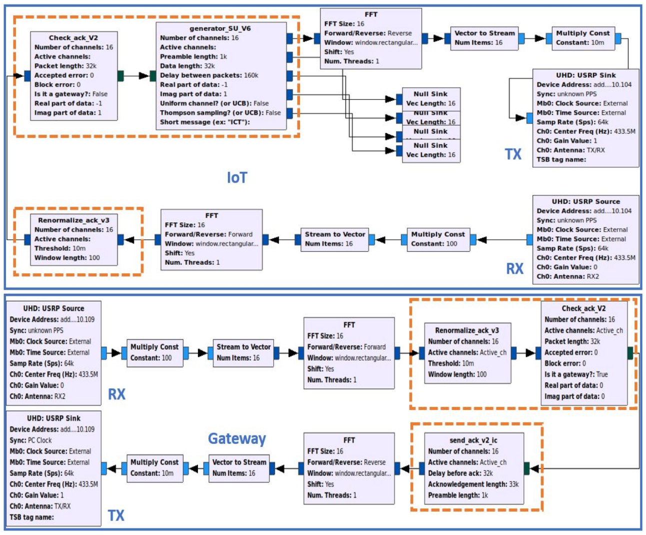


Figure 2: A data packet composed of a preamble, a field named UP/DOWN, and a user ID given by six blocks of QPSK symbols, B_k.

downlink packets, respectively. The ID user is defined by 6 blocks of QPSK symbols B_k for $k \in [1\ 6]$, which are assigned two possible constellation symbols. Then, a threshold is applied to the number of times that a symbol is received within each block B_k, so that a decision about the transmitted symbol is made. Hence a binary sequence of 6 bits are obtained, and consequently a total of 64 users can be supported [2].


Check_ack_V2
Number of channels: 16
Active channels:
Packet length: 32k
Accepted error: 0
Block error: 0
Is it a gateway?: False
Real part of data: -1
Imag part of data: 1

generator_SU_V6
Number of channels: 16
Active channels:
Preamble length: 1k
Data length: 32k
Delay between packets: 160k
Real part of data: -1
Imag part of data: 1
Uniform channel? (or UCB): False
Thompson sampling? (or UCB): False
Short message (ex: "ICT"):

FFT
FFT Size: 16
Forward/Reverse: Reverse
Window: window.rectangular...
Shift: Yes
Num. Threads: 1

Vector to Stream
Num Items: 16

Multiply Const
Constant: 10m

Null Sink
Vec Length: 16

Null Sink
Vec Length: 16

Null Sink

Null Sink
Vec Length: 16

UHD: USRP Sink
Device Address: add....10.104
Sync: unknown PPS
Mb0: Clock Source: External
Mb0: Time Source: External
Samp Rate (Sps): 64k
Ch0: Center Freq (Hz): 433.5M
Ch0: Gain Value: 1
Ch0: Antenna: TX/RX
TSB tag name:

Renormalize_ack_v3
Number of channels: 16
Active channels:
Threshold: 10m
Window length: 100

FFT
FFT Size: 16
Forward/Reverse: Forward
Window: window.rectangular...
Shift: Yes
Num. Threads: 1

Stream to Vector
Num Items: 16

Multiply Const
Constant: 100

UHD: USRP Source
Device Address: add....10.104
Sync: unknown PPS
Mb0: Clock Source: External
Mb0: Time Source: External
Samp Rate (Sps): 64k
Ch0: Center Freq (Hz): 433.5M
Ch0: Gain Value: 0
Ch0: Antenna: RX2

UHD: USRP Source
Device Address: add....10.109
Sync: unknown PPS
Mb0: Clock Source: External
Mb0: Time Source: External
Samp Rate (Sps): 64k
Ch0: Center Freq (Hz): 433.5M
Ch0: Gain Value: 0
Ch0: Antenna: RX2

Multiply Const
Constant: 100

Stream to Vector
Num Items: 16

FFT
FFT Size: 16
Forward/Reverse: Forward
Window: window.rectangular...
Shift: Yes
Num. Threads: 1

Renormalize_ack_v3
Number of channels: 16
Active channels: Active_ch
Threshold: 10m
Window length: 100

Check_ack_V2
Number of channels: 16
Active channels: Active_ch
Packet length: 32k
Accepted error: 0
Block error: 0
Is it a gateway?: True
Real part of data: 0
Imag part of data: 0

UHD: USRP Sink
Device Address: add....10.109
Sync: PC Clock
Mb0: Clock Source: External
Mb0: Time Source: External
Samp Rate (Sps): 64k
Ch0: Center Freq (Hz): 433.5M
Ch0: Gain Value: 1
Ch0: Antenna: TX/RX
TSB tag name:

Multiply Const
Constant: 10m

Vector to Stream
Num Items: 16

FFT
FFT Size: 16
Forward/Reverse: Reverse
Window: window.rectangular...
Shift: Yes
Num. Threads: 1

send_ack_v2_ic
Number of channels: 16
Active channels: Active_ch
Delay before ack: 32k
Acknowledgement length: 33k
Preamble length: 1k

Figure 3: The GRC designs for the IoT device and the gateway are described at the top and bottom of the figure, respectively. The blocks corresponding to our framework are higlighted (in orange boxes) within the flowgraphs.

3 Results

We evaluate our demo in a scenario with a clear line of sight (LOS) by placing two IoT users and a gateway, all of them working at a carrier frequency of 433.5 MHz. Each IoT user embedding a UCB algorithm [3] is able to select among four frequency channels and transmit packets of roughly 0.5 seconds following a LoRa standar. In Fig. 3, the implemented GNU radio companion (GRC) designs are depicted, where a sample rate of 64 kbps, and 4×10^3 symbols for each block B_k are configured. An FFT block of 16 bins is employed to a provide a channel selection mechanism with four channels indexed as 2, 5, 9, and 12. In this example, one of the IoT user is set to use the channels 9 and 12, whereas a second IoT user is able to choose among the four channels. The obtained results in Fig. 4 shows that the second IoT user learns to select the available channels 2 and 12 (curve in blue), meaning that the gateway is able to handle the incoming packets by replying with the corresponding ACK packets. On the other hand, a gap is observed between the number of trials and successful transmission due to the collisions involved in the learning process.

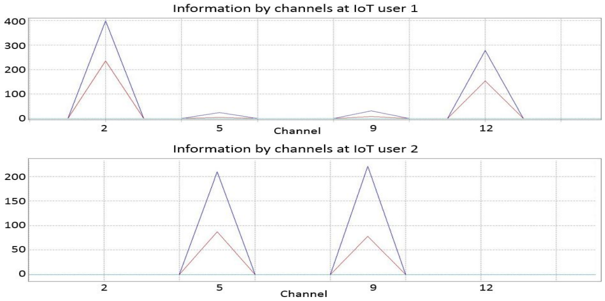


Figure 4: Number of times a channel is selected (curve in blue) and the number of successful transmissions (curve in red) for each channel and IoT users.

4 Conclusion

We have presented a GNU radio implementation that recreates an IoT network for the evaluation of access policies based on reinforcement learning approaches. Our framework introduces a packet structure to handle a large number of IoT users that may incorporate MAB algorithms. Finally, the experimental results show that a UCB approach improves the performance of the IoT user. Furthermore, the modular design of the proposed framework allows the evaluation of any novel access policy, as well as the incorporation of other physical layers.

References

[1] L. Besson, R. Bonnefoi, and C. Moy, "GNU Radio Implementation of MALIN: "Multi-Armed bandits Learning for Internet-of-things Networks"," in *To appear in 2019 IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2019.

[2] R. Kumar, S. Darak, A. Sharma, and R. Tripathi, "Two-stage decision making policy for opportunistic spectrum access and validation on USRP testbed," *Springer: Wireless Networks*, vol. 24, pp. 1509–1523, 2018.

[2] The length of the fields can be adjusted to support more users. In fact, our implementation allows to divide the blocks B_k into chunks of symbols so as to convey more bits. Furthermore at higher sample rates, more symbols per block B_k are available.

[3] For a more detailed illustration of the implemented UCB algorithm, the reader may refer to [1].