Poster: Extended LoRaSim to Simulate Multiple IoT Applications in a LoRaWAN

Muhammad Omer Farooq
Nimbus Centre for Embedded Systems Research
Cork Institute of Technology, Cork, Ireland
omer.farooq@cit.ie

Dirk Pesch
Nimbus Centre for Embedded Systems Research
Cork Institute of Technology, Cork, Ireland
dirk.pesh@cit.ie

Abstract

LoRa/LoRaWAN is a popular low-power wide-area networking (LPWAN) technology enabling Internet of Things (IoT) applications. LoRa/LoRaWAN supports a range of communication settings for a large number of nodes. Before proceeding with the deployment of IoT applications on top of a LoRaWAN network, it is sensible to conduct simulation-based studies to optimise the design of a LoRaWAN network for the IoT applications under consideration. LoRaSim is currently the most popular simulator for LoRa/LoRaWAN. However, LoRaSim currently only supports network simulation with only a single IoT application, while real deployments typically support multiple applications through a single LoRaWAN network. In order to address this, we have extended LoRaSim to simulate multiple concurrent IoT applications. To demonstrate the usefulness of our extension, we present a simulation-based case study along with results obtained through the extended simulator.

1 Introduction

Recently, low-power wide-area networking (LPWAN) technologies, such as Long Range (LoRa) [1], SigFox [2], and Weightless [3] among others have emerged to support a wide range of Internet of Things (IoTs) applications. LoRa/LoRaWAN is one of the most popular among the LPWAN technologies due to its openness and open source software support. Moreover, to support a wide operating range, LoRa provides a number of communication settings based on physical layer parameters that can be customized, such as spreading factor (SF), Bandwidth (BW), transmission power (TP), and code rate (CR). A simple LoRaWAN network consists of a gateway, a number of LoRa nodes, and a server. However, a complex LoRaWAN network may comprise of multiple gateways. Usually, LoRa nodes are connected to the gateway in a star topology, and the gateway is connected to the server. A simple Aloha protocol is used for packet transmission in LoRaWAN networks.

Before deploying a physical network, network planners typically evaluate a particular network design, protocol parameters and different applications using a network simulator. A network simulator is not only a relatively inexpensive tool for design evaluation, but it also provides an early insight into different aspects that can affect a network design's performance. LoRaSim [4] is currently the most popular LoRa/LoRaWAN simulator. However, LoRaSim currently only allows to simulate LoRaWAN networks with a single application [4]. However, real deployments support multiple IoT applications, such as smart metering, smart parking, and street lighting using a single LoRaWAN network. Different IoT applications have different data generation patterns and quality of service requirements. Therefore, to accommodate the study of such multi-application use cases, we present an extension of LoRaSim that can simulate a LoRaWAN running multiple applications.

2 LoRaSim: A LoRa Network Simulator

LoRaSim [4] is a discrete-event simulator developed using the Python programming language's SimPy package. The simulator implements a radio propagation model based on the well-known log-distance path loss model. The sensitivity of a radio transceiver at room temperature w.r.t. different LoRa Spreading Factors (SFs) and Bandwidth (BWs) settings is calculated using Equation 1. In Equation 1, -174 represents the thermal noise power across 1 Hz of BW in dBm. Receiver bandwidth is BW, receiver noise figure is NF, and SNR is the signal-to-noise ratio.

$$S = -174 + 10log_{10}(BW) + NF + SNR \tag{1}$$

$$C(x,y) = O(x,y) \wedge C_{freq}(x,y) \wedge C_{sf}(x,y) \wedge C_{pwr}(x,y) \wedge C_{cs}(x,y) \tag{2}$$

LoRaSim infers the collision of two packets x and y from the following parameters: reception overlap '$O(x,y)'$, carrier frequency '$C_{freq}(x,y)'$, spreading factor 'C'_{sf}, transmit power '$C_{pwr}(x,y)'$, and timing '$C_{cs}(x,y)'$. The simulator considers that packets x and y have collided, if eqn. 2 evaluates to true. Further information on communication range, path loss, and collision behavior can be found in [4]. LoRaSim generates data packets using a Poisson process and simulates the following LoRa communication settings.

International Conference on Embedded Wireless Systems and Networks (EWSN) 2018
14–16 February, Madrid, Spain

ISBN: 978-0-9949886-2-1

SN^0 This setting uses the following parameters: SF12, BW = 125 KHz, and CR = $\frac{4}{8}$.

SN^1 Similar to SN^0, however randomly chooses between three different transmit frequencies (860, 864, and 868) MHz.

SN^2 SF6, BW = 500 KHz, and CR = $\frac{4}{5}$.

SN^3 Uses optimized settings per node based on the node's distance from the gateway.

SN^4 SF12, BW = 125 KHz, and CR = $\frac{4}{5}$. It is the recommended LoRaWAN setting.

SN^5 This setting is similar to SN^3, however it also optimizes the transmit power.

3 Extended LoRaSim

Our extended LoRaSim is based on LoRaSim. It differs from LoRaSim in the following aspects:

- Multiple IoT applications can be simulated in the same LoRaWAN network. For each application, a user can specify the application's data generation model along with data packet size. Supported data generation models include the following: exponentially distributed traffic (Poisson process), randomly distributed traffic, and periodic traffic. However, other generation models can also be added.
- A user can specify the number of nodes corresponding to each application.

To use the extended LoRaSim, a user has to prepare a simulation configuration file. A record in the file comprises of the following information: *no_of_nodes, data_distribution_id, pkt_generation_rate,* and *pkt_size*. The interpretation of *pkt_generation_rate* is dependent on *data_distribution_id*. *data_distribution_id* can take one of the following values: *'-P'* for periodic data generation, *'-E'* for exponentially distributed data generation, and *'-R'* for random data generation. Typically, IoT applications generate data packets periodically or randomly, therefore our LoRaSim extension supports the mentioned data packet generation models. *pkt_generation_rate* is specified in milliseconds (ms), for *data_distribution_ id* corresponding to *-P*, *-E*, and *-R* it represents periodic packet generation interval, mean packet generation interval, and upper limit on the time after which a packet should be generated, respectively. A single record in the configuration file corresponds to an application. The configuration file is used as an input to our extended LoRaSim. At the end of a simulation run, the simulator reports packet delivery ratio (PDR) corresponding to each application and total energy consumption. To run the simulator the following command line arguments are required: *configuration file, LoRa communication setting number to simulate, total simulation duration,* and *full stack collision check indicator*. The number corresponding to SN^0, SN^1, SN^2, SN^3, SN^4, and SN^5 LoRa communication settings are 0, 1, 2, 3, 4, and 5 respectively.

Here we present a couple of LoRa network simulation case studies using our extended LoRaSim. The configuration files for case study are shown in Figure 1. We

Configuration File I	Configuration File II
50 –P 86400000 80	50 -P 86400000 80
100 -E 3600000 60	100 -E 3600000 60
50 -R 43200000 40	50 -R 43200000 40
	100 -P 3600000 20
	50 -E 1800000 30
	100 -R 3600000 10

Figure 1. Simulation Configuration Files

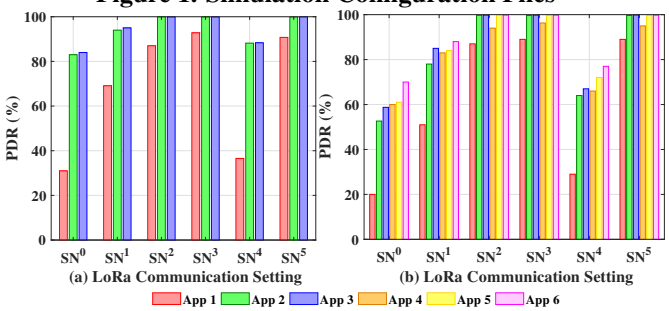

Figure 2. Simulation Results

simulated a LoRa network with communication settings SN^0, SN^1, SN^2, SN^3, SN^4, and SN^5. Each simulation duration was 1 month.

Figure 2 (a) and Figure 2 (b) show the Packet Delivery Ratio (PDR) for different applications belonging to simulation case study I and II respectively. Among the evaluated communication settings, mostly SN^0 and SN^4 demonstrate poor performance. Packet air time corresponding to SN^0 is higher compared to other evaluated settings. As LoRa uses Aloha for channel access, a higher packet air time results in a higher number of collisions, hence poorer performance. SN^4 performs similar to SN^0 as the only difference between the two settings is the code rate. Among the different simulated applications, applications that generate periodic data traffic demonstrate poorer performance. As there are a number of nodes in the network periodically generating data packets with the same interval, they experience a higher probability of packet collision. To lower the collision probability in such scenarios, the extended LoRaSim randomly delays the transmission of data packets corresponding to the periodic *data_generation_id*.

4 Conclusions

To accommodate the study of multi-application use cases in LoRaWAN networks, we extended LoRaSim so that it can simulate a LoRaWAN running multiple applications. Through an example simulation study, we showed and discussed the usage and functionality of the extended LoRaSim.

Acknowledgments - this research has received support from Science Foundation Ireland under Grant Number 13/RC/2077.

5 References

[1] LoRa Alliance. https://www.lora-alliance.org/. Last accessed: 07th January, 2017.

[2] SigFox. https://www.ekito.fr/people/connecting-things-to-the-internet-with-sigfox/. Last accessed: 07th January, 2017.

[3] Weightless. http://www.weightless.org/. Last accessed: 07th January, 2017.

[4] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso. Do LoRa Low-Power Wide-Area Networks Scale? In *Proceedings of the* 19th *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '16, pages 59–67, 2016.