LoRaCity: A Simulation and Planning Tool for the Deployment of Citywide LoRa-based networks

Soledad Escolar, Fernando Rincón, Julián Caba, Jesús Barba, Félix J. Villanueva, Maria J. Santofimia, David Villa, Xavier del Toro, Juan Carlos López [1]

Resumen— **Long range wireless communication has become a very suitable candidate technology for data transmission in some scenarios of Internet of Things (IoT), due mainly to its long-range capacity and energy efficiency. However, its application to cover large surfaces, such as small or medium sized cities, has not been deeply investigated and presents uncertainties in terms of performance, cost, and coverage. In this sense, one of the fundamental processes in the design of the network is its planning and simulation before the physical deployment on the surface to be covered. This work proposes to bridge this gap by means of the development of a simulator called LoRaCity, based on LoRa technology, that allows to evaluate a variety of LoRa network configurations on the map of the city that is intended to cover, and whose simulation results may support the process of planning these networks over medium and large areas.**

Palabras clave— **Long range communication; LoRa; Simulator; City-wide Network; Internet of Things.**

I. Introduction

Ubiquity in Internet of Things (IoT) demands wireless communication technologies adapted to the applications needs and the device constraints, that are characterized by the need to optimize their energy consumption, since they are generally standalone devices that base their operation on the use of limited-capacity batteries. Traditionally, short-range communication technologies, such as ZigBee or Bluetooth, have been used to provide wireless connectivity to the sensing devices. However, this type of solutions pose several drawbacks: first, the devices may be separated by a distance of only a few metres. For example, for ZigBee the theoretical maximum distance is 100 metres and for Bluetooth is just 10 metres; second, the high cost of deployment due to the need to install a larger number of devices in order to cover wide areas; and third, when tree-based network topologies are used, the devices closer to the base station drain their batteries much faster than devices at the periphery, due to the need to forward all packets generated in the network towards the base station, which makes the lifetime of the network depends directly on the lifetime of the devices near to base stations. An alternative to short-range communication is long-range communication, which enables Low-Power Wide-Area Networks (LP-WAN), characterized by long-range links, which make them ideal for some IoT applications.

Long-range technologies, such as LoRa, have been already employed for connecting devices in smart cities. For example, in [1] the authors present the RIGERS project, where two experiments for building monitoring were implemented in two districts of Bologna. One experiment consisted in deploying a total of 25 LoRa devices and a gateway to cover the surfaces of 0.9×1.8 km and a square area of side 0.6 km. These LoRa networks adopted a star-topology, where the end-devices transmitted data every five minutes to the gateway, which forwarded the collected data to the control center via a 3G network. The results demonstrated that the maximum distance from the gateway to the device is 2390 meters. In [2] another large-scale smart city demonstrator was conducted at the University of Lille, in France. The experiment consisted in 46 end-devices based on Raspberry PI with a Libelium LoRa module that transmitted data towards an only gateway which, in turn, forwarded data to a network server using a 4G connection. The results shown that LoRa technology provides a good performance to cover a maximum distance of 1.2 km and poor signals were mainly due to the presence of large buildings.

As observed in these two demonstrators, it is the general case that, the planning of a LoRa network requires to undertake decissions related to the architecture design and the physical deployment, that are targeted to the scenario to be addressed and, therefore, they are manually implemented without the support of tools that help to their automation. Among these decissions are, for instance, the network topology, the location of devices, amount of devices and distances between them. In this paper we propose LoRaCity, a simulation and planning tool that helps the technicians to take strategic decisions about the LoRa network, based on input parameters and simulation results, and whose ultimate goal is to contribute the automatization of the efficient planning of medium/large infrastructures that need to communicate by using LoRa technology. As far as the authors know, this is the first work addressing this objective. The remainder of this paper is organized as follows. We describe in Section II the specifications of LoRa technology and we review the related works in Section III. Section IV presents the system model and Section V presents our simulation tool LoRaCity. In Section VI we provide the results of our simulator to plan the deployment of the LoRa

[1]School of Computing Science, University of Castilla-La Mancha, Ciudad Real, Spain, e-mail: {`soledad.escolar, fernando.rincon, julian.caba, jesus.barba, felix.villanueva, mariajose.santofimia, david.villa, xavier.deltoro, juancarlos.lopez`}`@uclm.es`

network over several cities. Finally, in Section VII we draw the main conclusions and make suggestions for further research.

II. Background

LoRa (Long Range) [3], [4] is one of the enabling technologies for LP-WANs [5], which are characterized by a long range communication, a low data rate and a low energy consumption. LoRa is especially targeted for applications that need to send small amounts of data a few times per hour over long distances (in the order of kilometers), which fits well with very diverse outdoor monitoring IoT scenarios, as for instance, smart agriculture. The work [6] compares LoRa against other enabling technologies for LP-WAN as are SigFox and NB-IoT. Table I presents a brief comparison of these technologies.

Tabla I: LP-WAN Technologies Comparison.

Features	**Sigfox**	**LoRa**	**NB-IoT**
Range	10 km (urban) 40 km (rural)	5 km (urban) 20 km (rural)	1 km (urban) 10 km (rural)
Data rate	0.1 kbps	0.3-27 kbps	200 kbps
Max. payload	12 bytes (up), 4 bytes (down)	243 bytes	1600 bytes
Messages/ day	140 (up), 8 (down)	Unlimited	Unlimited
Modulation	BPSK	CSS	QPSK
Frecuency Bands	ISM Bands: 868 MHz (Europe), 915 MHz (North America), 433 MHz (Asia)	ISM Bands: 868 MHz (Europe), 915 MHz (North America), 433 MHz (Asia)	Lic. LTE Frequency Bands
Bandwidth	100 Hz	125, 250, 500 kHz	200 kHz
Type	Half-duplex	Half-duplex	Half-duplex
Cost	Free	Free	€500M/MHz

LoRa operates in unlicensed ISM frequency bands that depend on the region in which a specific LoRa solution is deployed. In Europe, LoRa operates in the 863 to 870 MHz band and in United States it operates in the 902 to 928 MHz frequency band. The bandwidth, in turn, depends on the frequency bands. For higher bandwidths the data rates are higher and the transmission times are lower. LoRa divides the band in different sub-bands or channels for uplink (messages from the end-device to an application running on a server) and downlink (a message from the server to the end-device) and the number of channels depends on the specific local regulations.

As a physical layer, LoRa standardizes a set of parameters. LoRa uses a proprietary modulation technique derived from Chirp Spread Spectrum (CSS) technique, where each bit of payload information is represented by multiple chirps of information. The spreading factor (SF), is defined as the number of chirps to represent a symbol, between 7 and 12, where a larger SF implies a longer transmission time and a longer communication range. An SF i allows to send 2 times more bytes than an SF $i+1$ in the same time or, alternatively, allows to reduce the time approximately to the half for a same payload. The transmission power ranges between 5 and 23 dBm, where higher power increases the energy consumption. The coding rate (CR) is the error correction coding given as A/B, where A is the data block length (A=4) and B is the codeword length (B $\in$ [5,8]); available rates are 4/5, 4/6, 4/7 and 4/8. A higher CR implies a larger overhead but a higher reliability of communication. The data transmission rate is the amount of data transmitted by unit of time (bps) and it depends on the SF, bandwidth BW, and CR. It is computed as $SF \times \frac{BW}{2^{SF}} \times CR$ and it ranges between 0.3 and 27 kbps. In practice, each application customizes its transmission mode by defining its SF, BW and CR. For example, the combination SF=12 and BW=125 kHz provides the maximum range and the slowest data rate, while a SF=7 and BW=500 kHz provides the minimum range, the fastest data rate and the lowest consumption. LoRa also specifies the packet format, which is composed of three elements: a preamble (8 symbols by default), an optional header, a variable-length payload and a CRC. The preamble enables to synchronize the receiver with the incoming data; the header, if exists, includes metadata to inform the CR, the payload length and payload CRC presence. The payload includes the data coded at the CR specified.

An application using LoRa may compute in advance the time of transmission for their messages as [7]:

$$T_{packet} = T_{preamble} + T_{payload} \tag{1}$$

where $T_{preamble}$ and $T_{payload}$ correspond, respectively, to the times to send the preamble and the payload, specifically:

$$\begin{aligned} T_{preamble} &= T_{sym} \cdot (NS_{Pre} + 4.25) \quad (2)\\ T_{payload} &= T_{sym} \cdot NS_{Pl} \quad (3) \end{aligned}$$

where $T_{sym} = \frac{2^{SF}}{BW}$ is the time to send a single symbol, and NS_{Pre} and NS_{Pl} are the number of symbols in the preamble and payload, respectively. The preamble is followed by a Start of Frame Delimiter (SFD) that occupies 4.25 symbols. In turn, NS_{Pl} is computed as:

$$8 + \max\left(0, \left\lceil \left(\frac{8PL - 4SF + 28 + 16 - 20H}{4(SF - 2DE)}\right)(CR + 4) \right\rceil\right) \tag{4}$$

with $H = 1$ if the header is present (otherwise, $H = 0$) and $DE = 1$ if the low data rate optimization is enabled (otherwise, $DE = 0$).

A LoRa gateway may cover several kilometers and serves up to potentially thousands of end-devices. In order to make a fairness sharing of the channel among the radio-based devices, the governments have regulated, among other parameters, the maximum duty cycle (DC), i.e. the fraction of time during which a device using unlicensed bands can occupy a channel. In Europe, the ETSI EN300.220 standard establishes that a DC may range between 0.1% and 10% in each sub-band. This means that, if a LoRa device (either end-device or gateway) is setup with a DC=1% then it can transmit up to 36 seconds per hour and sub-band. The maximum time in which the device can be using a sub-band is called $T_{airtime}$. Note that such constraint impacts both on the length of the messages to transmit and on the wait time among messages. Thus, the maximum number of packets n_{packet} transferred during $T_{airtime}$ for packets of duration T_{packet} is:

$$n_{packet} = \left\lfloor \frac{T_{airtime}}{T_{packet}} \right\rfloor \tag{5}$$

and the time required to wait between packets T_{wait} is:

$$T_{wait} = \frac{1}{\sum_{i=1}^{n} \frac{DC(i)}{T_{packet}}} \tag{6}$$

where n is the number of channels and DC(i) represents the duty cycle as a fraction $\in [0, 1]$. Note that this limitation of the duty cycle may reduce not only the suitability of LoRa for some IoT applications (e.g. the ones that require real-time data transmission at high frequencies or the transmission of huge amounts of data) but also may reduce the amount of devices that a gateway can simultaneously support.

III. Related Works

Based on the equations of Section II, many works in the literature have described analytical studies and results for different LoRa network configurations in terms of packet delivery ratio, maximum number of end-devices supported by a gateway (scalability), coverage range, network density and collisions [8], [9], [10]. The simulation tools have proved their efficiency and accuracy to enable the automatization of this theoretical process and supported the experimentation of a great variety of network configurations as well as providing interesting features as the reproducibility of experiments and graphical interfaces. This section describes three of the most popular LoRa simulators, LoRaSim [11], FREE [12] and FLoRa [13], emphasizing their main advantages and weakness. The three simulators emulate the LoRa physical layer and link-level behaviour. They enable multiple configurations of LoRa devices (e.g. SF, CR, BW) and networks set-up (e.g. number of end-devices and gateways, payload length, and transmission period). All of them are based on discrete events and provide multi-platform support (Windows, Linux and MacOS). Table II shows a brief comparison of these simulators.

Tabla II: Simulator Comparison.

Features	LoRaSim[11]	FREE[12]	FLoRa[13]
Language	Python 2		C++
Platform	Windows, Linux, MacOS		
Indoor dist. Outdoor dist.	300 m. 3000 m.	300 m. 3000 m.	480 m. 9800 m.
Deployment	Random		
end-devices	Unlimited		
gateways	1-6, 8, 24	1	Unlimited
Retx.	No	Yes	Yes
Energy analysis	No	Yes	Yes

LoRaSim is a simulator[1] aimed at simulating different LoRa network settings. Their authors then question the scalability of LoRa networks and report that, for a typical smart city deployment (with a selected conservative transmission settings and one only gateway) with nodes sending 20 bytes-length packets each 16 minutes, a gateway can support 120 end-devices per 3.8 hour, which is insufficient for future IoT deployments [11]. In FREE, the authors updated LoRaSim to consider a packet error model, the imperfect orthogonality of spreading factors, the fading impact, and the duty cycle limitation at both, the devices and the gateway. FREE supports bidirectional communication by adding the downlink capability and the retransmission strategy in case of confirmable uplink transmissions. FREE also extends the energy consumption profile from LoRaSim to consider the consumed energy at the reception time. However, it loses the possibility that LoRaSim has to simulate several gateways. FLoRa has currently implemented a single communication channel, so that all the nodes of the network have the same transmission frequency, being a very limiting factor to be able to take advantage of the capabilities offered by this type of networks.

One important lackness of these simulators is that they offer scarce or null control on the network deployment. Although an user may specify the number of end-devices and gateways, it is not possible to control their locations, distances, and the target field where they will be installed, which is an impediment to the efficient network deployment. Furthermore, we have not found evidence of any simulator that allows loading maps to perform simulations. It is the purpose of our simulator LoRaCity to provide a discrete event simulator that enable mesh deployment based on distances among devices, possibility to load maps for simulation, detailed calculation of the energy consumption of each node and computation of a great variety of output statistics.

IV. System Model

We address the problem of determining a LoRa-based connection infraestructure for a certain small-/medium city that is represented by means of a map. Our goal is to find the minimal deployment that is required to cover completely the city with LoRa stations. To this end, we need to determine the number of LoRa end-devices and the number of LoRa static gateways that are neccessary in order to all end-devices may transmit effectively data towards a some gateway which, in turn, may forward the collected data to a network server or cloud facility to store, process and use the data for multiple purposes. We denominate this problem the Minimum Citywide LoRa Deployment (MCLD).

The LoRa infrastructure to be installed over the city pursues surrounding monitoring purposes. Subsequently, each LoRa end-device is equipped with

[1]Available at: `http://www.lancaster.ac.uk/scc/sites/lora/`

several sensors to monitor parameters of interest (e.g. temperature, humidity, air quality) with a certain periodicity τ and a LoRa radio that transmits a number of packets n_{packet}. This restriction guarantees that the end-devices do not perform an over-occupation of the channel and meet the limitation of the duty cycle mentioned in Section II, i.e. DC should be between 0.1% and 10%. Note that if the monitoring frequency τ is larger than the transmission frequency, then the application needs to apply data aggregation techniques over the values collected to reduce the number of packets to transmit to n_{packet}.

Let us consider a LoRa radio range of $r \leq LoS$ kilometers for both LoRa end-devices and gateways. LoS stands for Line of Sight of a LoRa transmitter, i.e., a straight line along which an observer has unobstructed vision. The maximum distance r between an end-device and a gateway is the one that ensures their connectivity. Note that, in order to preserve the LoS, the obstacles between the end-devices and gateways should be avoided. Note also that, however, this requirement could be difficult to find in urban environments and the network planner is then the responsible of locating the devices in positions obstacle-free (e.g. top of the roofs). According to [14], a range of over 4 km was observed in a free LoS scenario.

The city to be covered is represented by a 2D map M that draws an irregular polygon of area A bounded by limits M=$\langle X_0, Y_0, X_1, Y_1\rangle$, where $\langle X_0, Y_0\rangle$ is the lower left corner and $\langle X_1, Y_1\rangle$ is the upper right corner of the map. The LoRa deployment on M follows a grid distribution, where the end-devices are regularly spaced at a maximum distance d. Under these constraints, we compute a deployment $D = \langle N, G\rangle$, where $N = n_1, n_2, \ldots, n_{k_1}$ is the set of k_1 LoRa end-devices and $G = g_1, g_2, \ldots, g_{k_2}$ is the set of k_2 LoRa gateways, where each end-device $n_i, i \in [1, k_1]$ and gateway $g_j, j \in [1, k_2]$ are located at coordinates $\langle x_i, y_i\rangle$ and $\langle x_j, y_j\rangle$, respectively, within the area A in map M. D is admissible if it holds the next two conditions:

- **C1: Connectivity condition** states that $\forall n_i \in N$, $\exists$ at least a $g_j \in G$ s.t. the distance between both is lower or equal than r, i.e. $\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \leq r$.
- **C2: Coverage condition** states that $\forall$ geographic point $z = \langle x_z, y_z\rangle$ within the area A in map M, $\exists$ at least one end-device $n_i \in N$ s.t. the distance between both is lower or equal than $d/2$, i.e. $\sqrt{(x_z - x_i)^2 + (y_z - y_i)^2} \leq \frac{d}{2}$.

Given a map M of a city of area A and given a LoRa range r, the MCLD problem is then formulated as follows: to find the minimum deployment $D = \langle N, G\rangle$, on the map M that minimizes k_1 and k_2 subject to **C1** and **C2** conditions hold.

V. The LoRaCity Simulator

We have developed a simulator named LoRaCity, which leverages FREE and LoRaSim simulators described in Section III. The purpose of LoRaCity is to support the process of MLCD computation and to assist the network administrator in the efficient planning of a LoRa network. The simulator has been developed in Python 3.3 and is available online in `https://github.com/UCLM-ARCO/LoRaCity`.

LoRaCity is composed of five main modules: *GUI*, to enter the simulation parameters; *Deployment*, to specify the set-up parameters of the LoRa network; *Maps*, for enabling the possibility of loading maps; *Energy*, for the calculation of energy consumption; and *Statistics*, to generate an advanced computation of output results. Figure 1 shows the major modules of the LoRaCity simulator, which are described in the next subsections.

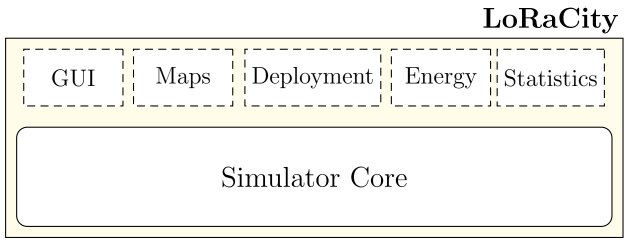

Fig. 1: Modules of LoRaCity Simulator

Most of the main functions of the core of the simulator have been reused from LoRaSim and FREE. Both simulators enable a deployment of a certain number of end-devices randomly on a circle of a certain radio, and where a set of gateways occupy fixed positions (only one in FREE, several in LoRaSim). The end-devices transmit periodically a LoRa packet with a fixed payload towards some gateway; gateway(s) that succesfully received the data packet sends back an ACK packet to the source to confirm its reception. If no ACK was received in some time interval the source proceeds to the packet retransmission. Gateways may also detect collisions, which occur when two or more packets overlaps at time; in this case, the gateway is unable to decode correctly one or several packets and no ACK is transmitted. From LoRaSim, we have adopted the possibility of deploying several gateways on the same deployment (not only limited to 24), which is essential to cover large distances. From the FREE simulator, the Bernoulli packet error model, and functions to cope with the limitations of the use of duty cycle have been reused. Since these functions have been developed in FREE to work with a single gateway, they have required to be adapted in LoRaCity to work with any number of gateways.

A. Graphical User Interface

Once LoRaCity is launched, a form enables the user to enter the configuration parameters of the network to simulate. We use PyQt5, a module of Python Qt graphical library. This form enables the user to select one of the maps previously saved in a *shapefile* format. Next, the user has to specify the maximum distance between each pair of end-devices

d and the maximum distance between a pair de gateways r (in kilometers). Note that these two parameters represent the minimum separation, in straight line, between any two nodes and between any two gateways, respectively. The rest of parameters requested correspond to the data transmission period, collision calculation strategy (0 for a simple check, 1 to consider the capture effect, and 2 to consider the SFs non-orthogonality and capture effect), SF and CR, length of payload, simulation time and simulation seed.

B. Maps

The possibility of loading maps for simulation has been developed through the use of *shapefiles* and the Python *geopandas* library. The user has to select a map that represents the geographic area where the LoRa network will be deployed. The format of the map is a *shapefile*, a simple, non-topological format used to store the geometric location and information of geographic entities, which can be represented by points, lines, or polygons. We have provided to our simulator the maps of Seville, Barcelona, Paris and Berlin cities, but note that any other city can be added just downloading its corresponding map in shapefile format. On the other hand, *geopandas* is an open source library to facilitate working with geospatial data; it provides various high-level operations, thus reducing the difficulty of use.

C. Deployment

LoRaCity computes a regular deployment $D = \langle N, G\rangle$ on the surface to be covered. The maximum number of end-devices k_1 and gateways k_2 is calculated as a result of the specified distance between nodes d and gateways r:

$$k_1 = \lfloor (X_1 - X_0)/d + 1\rfloor \times \lfloor (Y_1 - Y_0)/d + 1\rfloor \quad (7)$$
$$k_2 = \lfloor (X_1 - X_0)/r + 1\rfloor \times \lfloor (Y_1 - Y_0)/r + 1\rfloor \quad (8)$$

where the map is defined by $\mathrm{M}=\langle X_0, Y_0, X_1, Y_1\rangle$. For locating the nodes and gateways over a map M several functions have been developed. Unlike the areas handled by LoRaSim and FREE, which are fixed-size rectangles, maps in LoRaCity are polygons of variable size and irregular shapes. The *arange* function of the *numpy* library will create vectors with values evenly spaced at the desired distance between nodes and gateways. The *meshgrid* function will create two matrices with the grid distribution of end-devices and gateways. By using these functions LoRaCity computes a first approach for k_1 and k_2 values, which are not minimum and need to be adjusted. In order to do that, the simulator has to check: 1) the coordinates of all end-devices and gateways drop within any of the polygons that make up the map; and 2) there are no redundant gateways. A pair of coordinates is valid, either of an end-device or gateway, if it drops inside the map. Otherwise, if the coordinates of an end-device drop out the map, the end-device is simply removed, since there is nothing to cover. In case of a gateway drops outside the map, however, since they are strategic for the coverage and scalability, we decided not to reduce the number of them but instead force it to be reassigned to the nearest point within the map. Thus, the criteria followed is: a gateway is *valid* if its coordinates drop inside the map or, alternatively, if its coordinates drop outside the map but it has end-devices within its range of coverage. On the other hand, a gateway is *redundant* if all the end-devices within its radio are also covered by some other gateway, in whose case the gateway can be removed. Afer these checks we obtain the minimum number of end-devices k_1 and gateways k_2.

D. Energy

LoRaCity implements a detailed breakdown of energy consumption for each node in the simulation, differently to LoRaSim and FREE, which provide the total amount of energy for transmission and reception for the overall deployment. LoRaCity uses the parameters in Table III to perform the computations related to the energy consumption. Equation (9) refers to the total energy in transmission mode and Equation (10) is the energy wasted for reception of ACK packets per node.

$$E_{tx} = T_{packet} \cdot I_{tx} \cdot V \cdot n_{packet} \quad (9)$$
$$E_{rx} = T_{ack} \cdot I_{rx} \cdot V \cdot n_{ack} \quad (10)$$

The total energy consumption of a device is then calculated as the sum of the energy wasted in transmission and reception mode, i.e., $E = E_{tx} + E_{rx}$.

Tabla III: Symbols and parameters.

Param.	**Description**	**Unit**
V	Voltage of the node (end-device)	V
T_{packet}	Time of transmission of a packet	s
T_{ack}	Time of reception of an ACK	s
I_{tx}	Current in transmission mode	mA
I_{rx}	Current in reception mode	mA
n_{packet}	No. of total packets transmitted by a node	-
n_{ack}	No. of total ACKs received by a node	-
E_{tx}	Energy consumed in Tx mode per node	mJ
E_{rx}	Energy consumed in Rx mode per node	mJ

E. Statistics

After a simulation, LoRaCity saves in an understandable and readable format a vaste amount of information, including all simulation results (e.g. collisions, retransmissions, successfully received packets, energy breakdown per node, etc.). A part of these results are also displayed in graphical mode, such as the representation of the layout of end-devices and gateways on the map, and several pie graphs to present the collisions, retransmissions and received messages. Note that these three metrics do not capture the individual end-device performance but the performance of the network deployment as a whole.

The percentage of received messages is computed by taking the Data Extraction Rate (DER) [11], which is the ratio of received messages by some gateway to the total number of transmitted messages by all end-devices over the total simulation time. Note that the number of transmitted messages includes both original packets and retransmissions. The percentage of collisions is computed by using the ratio between the number of messages collided at some gateway to the total number of transmitted messages by all end-devices over the simulation time. Finally, the percentage of retransmissions is computed as the ratio between the original packets (one per node and period) to the total number of transmissions.

VI. Simulations

We have evaluated LoRaCity in terms of the minimum deployment that is required for completely covering cities of different areas, as well as the percentage of collisions, retransmissions, and successfully received packets under different network conditions and LoRa parameters. For our experiments we have used the available maps of Barcelona (101,9 km^2), Paris (105,4 km^2), Seville (140,8 km^2) and Berlin (891,8 km^2) cities. Finally, we have analyzed the results of the LoRaCity simulator to estimate the deployment costs of a LoRa network.

For the mentioned cities, Figure 2 and Figure 3 show the minimum number of end-devices k_1 and gateways k_2, respectively, that should be deployed keeping between the end-devices the minimum distances of $d = \{1, 2, 3, 4\}$ and between gateways the minimum distances of $r = \{2, 4, 6, 8\}$. Because of LoRaCity computes a grid deployment, such distances represent the minimum separation between any pair of adjacent devices/gateways. The larger is the distance between devices/gateways the minimum is the number of devices/gateways to install. Therefore, a gateway may theoretically cover all devices found within its coverage radio r.

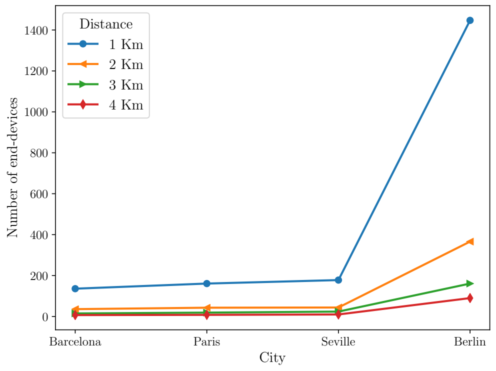

Fig. 2: No. of end-devices k_1 to deploy on several cities for different d

LoRaCity shows the minimal deployment computed on the map of the city selected. Figure 4 shows a minimal deployment, where the end-devices (black nodes) are separated a minimum distance of $d = 1$ km and the gateways (red nodes) are separated a minimum distance of $r = 3$ km. The coverage radius of each gateway is r, and the area covered by each one is represented by means of a red circumference. Note that this deployment has not redundant gateways and that some of them were relocated when needed. This minimal deployment guarantees the connectivity condition (**C1**) between any device and some gateway, i.e. every device is within the coverage radius of at least one gateway. To guarantee the coverage condition (**C2**) between any point of the map and any some end-device, i.e. every point is whithin the coverage radius of at least one device, the distance between both should not exceed $\frac{d}{2}$.

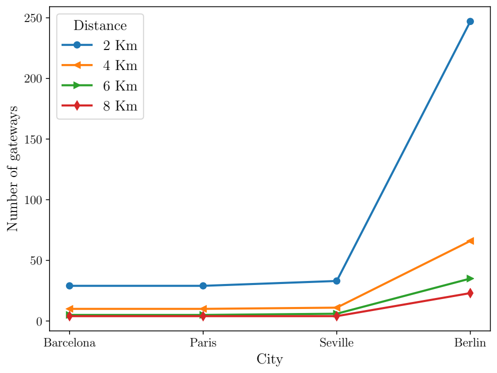

Fig. 3: No. of gateways k_2 to deploy on several cities for different r

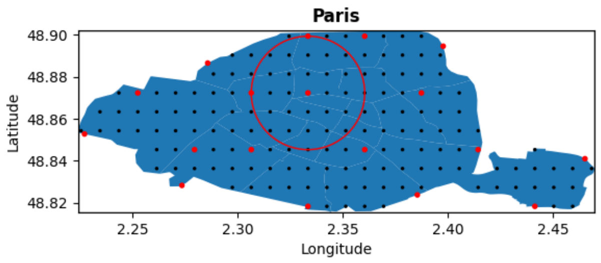

Fig. 4: Minimal deployment for Paris city with $d = 1$ and $r = 3$ km

LoRaCity supports also the evaluation of different LoRa settings for a certain deployment, depending on the needs of the network planner. To illustrate this fact we evaluate, for the city of Paris with $k_1 = 161$ and $k_2 = 29$ (given $d = 1$ and $r = 2$ km), the impact of varying the $SF \in [7, 12]$ and analyze the network performance in terms of number of transmitted messages, successfully received messages and collisions. To this end, we simulate an application that generates a 20-bytes payload packet every 10 minutes, with CR=4/5 and BW=125 kHz. As observed in Figure 5, 100% of the packets transmitted were succesfully received by, at least, one gateway. Note that the number of collisions increases with the SF, since the length of the packets is longer and therefore the transmission time.

Finally, we show in Figure 6 the energy breakdown into transmission and reception mode, for end-devices in the deployment computed for Paris city with $d = 3$ km and $r = 4$ km, SF=7, CR=4/5 and BW=125 kHz, that results into $k_1 = 19$ end-nodes and $k_2 = 9$ gateways. The energy consumption in transmission mode depends on the number of packets transmitted, both the original ones and retransmissions. The energy consumption in reception mode is due to the reception of ACKs; the number of ACKs received depends on, in turn, the number of gate-

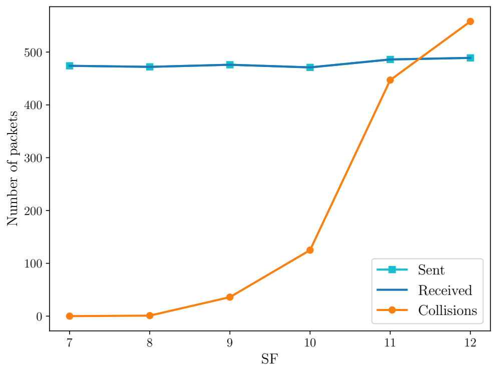

Fig. 5: Number of packets transmitted, received and collisions for different SF in Paris city with different SF

ways that are actually covering the end-device. Note that, although we removed the redundant gateways, it could still happen that a single device is within the coverage radius of several gateways simultaneously.

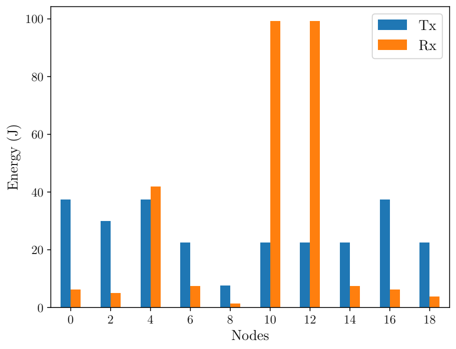

Fig. 6: Energy breakdown into transmission and reception mode, for the minimal deployment computed for Paris, with $d = 3$, $r = 4$km

VII. Conclusion and Future Works

We present LoRaCity, a tool to contribute the automatization of the efficient planning of medium/large infrastructures that need to communicate by using LoRa technology. Our proposal leverages LoRaSim and FREE simulators and extends their functionalities for providing multiple capabilities: graphical interface, loading of maps as physical scenarios where the network will be deployed, computation of the minimum deployment required to guarantee connectivity and coverage, visualization on the map as a grid, detailed computation of the energy consumption per node and statistics on the simulation results. LoRaCity may help to the network planner during the process of analysis and network design, and answer questions related to the number of devices that are required, their physical locations or the best LoRa configuration to use. As further research we plan to add more flexibility to LoRaCity by means of a larger number of type of deployments, as for instance, ad-hoc deployments, random, linear, and hybrid.

Acknowledgment

This research has been partially supported by the EU's Horizon 2020 programme under project SHAPES (GA Nº 857159) and Grant TALENT (PID2020-116417RB-C44) funded by MCIN/AEI/10.13039/501100011033.

Referencias

[1] Gianni Pasolini, Chiara Buratti, Luca Feltrin, Flavio Zabini, Cristina De Castro, Roberto Verdone, and Oreste Andrisano, "Smart city pilot projects using lora and ieee802.15.4 technologies," *Sensors*, vol. 18, no. 4, 2018.

[2] Marine Loriot, A. Aljer, and I. Shahrour, "Analysis of the use of lorawan technology in a large-scale smart city demonstrator," *2017 Sensors Networks Smart and Emerging Technologies (SENSET)*, pp. 1–4, 2017.

[3] LoRa Alliance, "LoRa," `https://www.lora-alliance.org`, May 2019.

[4] LoRa Alliance, "LoRaWAN Specification V1.0.3," `https://lora-alliance.org/sites/default/files/2018-07/lorawan1.0.3.pdf`, July 2018.

[5] Stephen Farrell, "Low-Power Wide Area Network (LPWAN) Overview," RFC 8376, RFC Editor, May 2018.

[6] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1 – 7, 2019.

[7] SEMTECH, "LoRa Modem Design Guide," `https://www.semtech.com/uploads/documents/LoraDesignGuide_STD.pdf`, July 2013.

[8] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the Limits of LoRaWAN," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, Sept. 2017.

[9] Martina Capuzzo, Davide Magrin, and Andrea Zanella, "Confirmed traffic in LoRaWAN: Pitfalls and countermeasures," in *17th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2018, pp. 1–7.

[10] Soledad Escolar, Fernando Rincón, Xavier del Toro, Jesús Barba, Félix Jesús Villanueva, María J. Santofimia, David Villa, and Juan Carlos López, "The PLATINO experience: A lora-based network of energy-harvesting devices for smart farming," in *XXXIV Conference on Design of Circuits and Integrated Systems, DCIS 2019, Bilbao, Spain, November 20-22, 2019*. 2019, pp. 1–6, IEEE.

[11] Martin Bor, Utz Roedig, Thiemo Voigt, and Juan Alonso, "Do lora low-power wide-area networks scale?," in *MSWiM '16 Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. Nov. 2016, pp. 59–67, ACM Press.

[12] K. Q. Abdelfadeel, D. Zorbas, V. Cionca, and D. Pesch, " *FREE* —Fine-Grained Scheduling for Reliable and Energy-Efficient Data Collection in LoRaWAN," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 669–683, 2020.

[13] M. Slabicki, G. Premsankar, and M. Di Francesco, "Adaptive configuration of lora networks for dense IoT deployments," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–9.

[14] G. Callebaut and L. Van der Perre, "Characterization of lora point-to-point path loss: Measurement campaigns and modeling considering censored data," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1910–1918, 2020.