An Overview On LoRaWAN Technology Simulation Tools

Mukarram A. M. Almuhaya 1,2, Waheb A. Jabbar1, 1,3, * Noorazliza Sulaiman1, and Sulaiman A. H.A1

[1] Faculty of Electrical & Electronic Engineering Technology, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia; almohia82@yahoo.com
[2] Sana'a Community College (SCC), Sana'a, Republic of Yemen; waheb@ieee.org
[3] Centre of Software Development & Integrated Computing, University Malaysia Pahang, 26300, Gambang, Pahang, Malaysia
waheb@ieee.org

Abstract. Low-Power Wide Area Networks (LPWAN) technologies are playing a pivotal role in the IoT applications owing to their capability to meet the keys IoT requirements, i.e., long-range, low cost, small data volumes, massive devices number, and low energy consumption. The creation of new public and private LoRaWAN networks necessitates the use of avoiding node limits and collision prevention measures. Designers of IoT systems confront difficulty in determining the scalability of a given technology, with an emphasis on unlicensed frequency bandwidth (ISM) transmission in densely populated locations. However, picking the best simulation software might be a challenge. To provide a conceptual overview of seven LoRaWAN simulation tools, this paper outlines their key characteristics and the sorts of experiments they support. LoRaWAN simulators, resource utilization, and performance evaluation are all covered in-depth in this report. Furthermore, we classify and compare the most important simulation tools for investigating and analyzing LoRa/LoRaWAN network emulators that have been developed recently. This article will be used to help other researchers decide whether LoRaWAN simulation tool is best for their specific requirements.

Keywords: IoT, LPWAN, LoRa, LoRaWAN, LoRa Simulation Tools.

1 Introduction

The LoRa is LPWAN technology which acquiring increasing attention from both academia and industry. To provide connectivity to a wide field of IoT devices, it is ideal for a low-power wide-area network's internet requirements(LPWAN). Chirp Spread Spectrum (CSS) modulation establishes a distinct radio layer from other types of wireless networks that use other modulations. Due to its high sensitivity, the LoRa CSS modulation allows for indoor transmissions with a range of several kilometres.[1]. To connect to the network server, the LoRaWAN device needs to be awakened and immediately transmits a packet to the network server through the gateway. This is similar to how ALOHA works [2]. The LoRaWAN systems that are spreading to various architectures are becoming increasingly complex, heterogeneous and pervasive. As such, it has proved difficult to research and design such systems. The lack of suitable modelling and simulation platforms to provide an end-to-end depiction of Connected IoT devices, i.e. from the basic IoT end-nodes to the cloud application and the fundamental networking infrastructure[3],[4]. Maximum node count constraints and collision avoidance algorithms are critical in the creation of new public and private LoRaWAN networks. The simulators mentioned in this article are used to model these issues. LoRaWAN techniques are equally important to model topological networks. The main challenges facing computer simulation environments are algorithms that support the positioning of gateways in combination with radio range modelling in urban environments. To assist practitioners and researchers overcome these challenges we propose this work as comparing between the commonly used LoRa/LoRaWAN simulation tools by highlighting their capabilities and features for enabling researchers to select the most suitable simulator based on their needs and programming skills. The platforms can model heterogeneous LoRa nodes (sensors, gateways, network servers, and so on) with quite well details (mobility, energy profile, scalability, and so on), diverse application logic and network connectivity models, as well. The proposed work is distinct from the existing literature, presents a comparative review of a selected seven driving LoRaWAN Simulators LoRaSim, NS3, Cupcurpon, FloRa, SimpleIoTSimulator and Mbed simulator, and gives a rundown of the most popular LoRa MAC network simulators and highlighted their most important features.

The rest of this paper is organized as follows: in Section II, we exhibit an overview of the LoRa/LoRaWAN technology. Then Section III surveys the available simulation tools to analyze LoRa/LoRaWAN performance. Discussion of the comparison in Section VI. Finally, we conclude the article in Section V.

2 LoRal/LoRaWAN

LoRa is a modulation technology patented and acquired by Semtech Corporation in 2012 for wireless communications [5]. LoRa is created to operate on a sub-GHz frequency, specifically on unlicensed bands such as 915 MHz, 868 MHz, or 433 MHz, in accordance with the regional area regulations. LoRa is a physical layer based on the Chirp Spread Spectrum (CSS) modulation, not similar to other modulations which using in other wireless networks [6]. LoRa was designed to be low-rate, low-power, and transmission with very long-range in line-of-sight or rural areas situations up to 10 or 20 km for outdoor, as a result of the higher sensitivity of LoRa CSS modulation, which enabling long distances connectivity [2]. Its low energy consumption, coupled with long-distance communication, makes LoRa be one of the potential candidates of LPWAN technologies for IoT applications [7]. As one of the most significant benefits of LoRa, the great receiver sensitivity is accompanied by an extremely wide communication connection budget. When utilizing LoRa modulation, typical values of SNR for 10 and 12 spreading factors are -20 dB and -15 dB, respectively, resulting in receiver sensitivities of -134 dBm and -129 dBm, according to the manufacturer. However, these

``

values are only somewhat equivalent to the average sensitivity of Wi-Fi or Bluetooth receivers, which is typically in the range of 40 dBm to 80 dBm. [6].

LoRaWAN networks can be generally utilized for fairly dense deployments with relaxed latency or reliability requirements. LoRaWAN offers many advantages in terms of low bit rate, power consumption, wide coverage, simplicity, and ease of management. While the LoRa modulation is proprietary, the LoRaWAN is an open standard being developed by the LoRa Alliance[8]. Each gateway in LoRaWAN networks receives messages from numerous end devices directly via a star topology. Transmitters employ TCP/IP protocols to communicate with network servers over a network connection. LoRaWAN's MAC uses Pure-ALOHA as its foundation. The LoRa specification defines three device classes that the end nodes must operate in, Class A, Class B and Class C. A LoRaWAN technology architecture defines three fundamental types of devices as follows[9]:

• End-devices (ED): These are devices that either take downlink (DL) traffic from the network server or generate uplink (UL) traffic for transmission to the gateways,

• Gateways (GW): These are the devices that demodulate LoRa communication and transmit it between the network server and the end devices in a wireless network. Wired or wireless access points link gateways to the Internet. The LoRaWAN Gateway is sophisticated, concurrently listening radio on several channels and delivering thousands of ENs simultaneously.

• Network Server (NS): The device which serves as the core backend of a LoRaWAN network, collecting traffic from all end-devices in the network and processing it on an application server

3 LoRa/LoraWAN Simulator Tools

Computer modelling and simulation is the proper method to enrich the exploration of systems performance and evaluate tactics for its functioning in imaginative or predictive approaches. A simulation model is a design that considers computing algorithms, physical and mathematical terms, and engineering formulas that summarise the behaviour and performance of a system's intangible world case studies. LoRa net-work simulation is more significant since it can be exploited without costly implementation or before the actual execution of the framework to design and evaluate a LoRa-based app. The field of LoRa offers highly specialized and freely available simulation tools. All these LoRa simulators have been developed and utilized in the literature for examining different LoRa scenarios, however, to the best of our understanding, none of the previous review studies compared in detail these simulation tools enough. Therefore, an overview of the most commonly used simulators to investigate Lo-Ra/LoRaWAN performance is presented in this section.

3.1 LoRaSim

LoRaSim has been developed based on SimPy as a discreet event simulator using Python to simulate, investigate, and analyze the LoRaWAN network scalability and collision functionality [10]. LoRaSim includes many Python scripts, the base of them are namely: loraDir.py, loraDirMulBs.py, directionalLoraIntf.py, and oneDirectional-LoraIntf.py, LoRaWAN.py, LoRaEnrgysim.py. The first script is for a lone gateway simulation, while loraDirMulBs.py is utilized to emulate several gateways (up to 24). DirectionalLoraIntf can emulate devices that are equipped with directional antennae and many networks, whereas oneDirectionalLoraIntf.py is for emulating gateways with directional antennae and many networks. A radio propagation model is implemented in LoRaSim depending on the well-known long-distance path loss model. The radio transceiver sensitivity at room temperature concerning various LoRa SFs and BWs settings is estimated. It also considers many related parameters such as the thermal noise power across, receiver bandwidth, noise figure, and SNR. Several packages are required for running LoRaSim smoothly, such as matplotlib, SimPy, and NumPy. LoRaSim offers a plots view of deployments with no graphical interface, as shown in Fig1. In contrast, it can show a data plot when users execute graphical code. Many improvements for LoRaSim were proposed to make it multipurpose[11-16] and support downlink due to the original version supported uplink only; thus, it can test scalability and energy consumption, and other matric performance. Many of the information is seen on the command line and exported to File-Name.dat, which can show its content data using other graphical programs such as Gnuplot, seaborn, Mplot, or others.

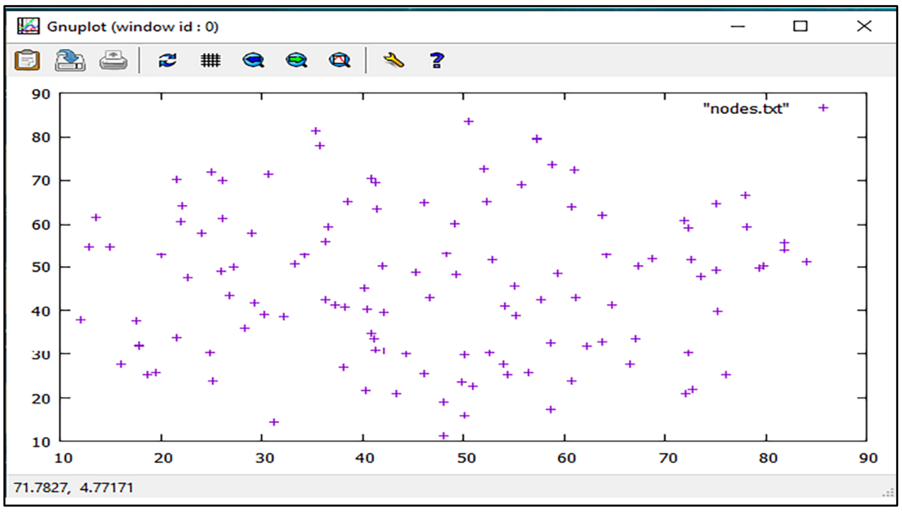
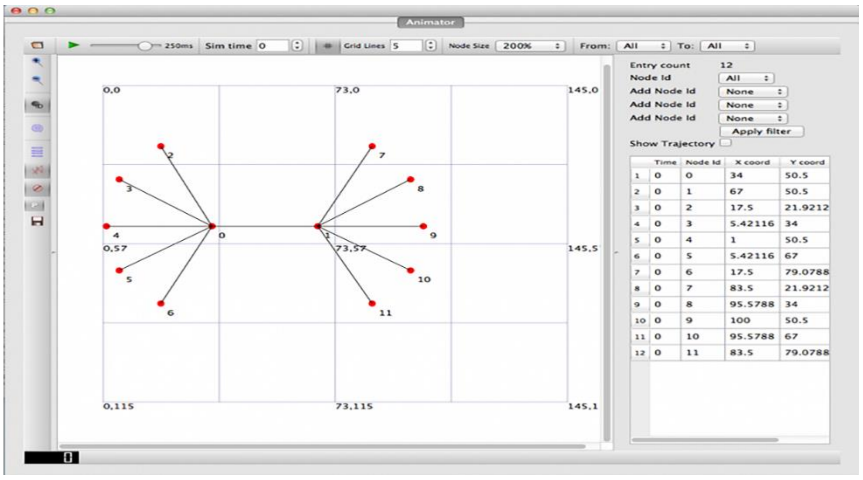


Fig. 1. LoRaSim network topology deployment

``

3.2 NS-3

Ns-3 is an open-source discrete-event network simulator that was initiated in mid-2006 [17, 18] and is still under heavy development now, written by C++ and Python. The ns3 simulator supports a wide variety of protocols such as Wi-Fi, LTE, IEEE 802.15.4, SigFox, LoRa, further networks and also implements an IP networking, supports both simulation and emulation using sockets that aim to academic and research [19]. The ns-3 can be executed with pure C++, and some simulation components can be written with Python. It is designed to be modular and can function in both graphical and command-line interfaces netanm for C++ and PyViz for Python. It also produces pcap tracks that can be used to debug. Standard software such as Wireshark [20] can be used to read trace files for network traffic analysis. The ns-3 offers a practical and well-structured setting with animation support using NetAnim, as shown in Fig 2. A LoRaWAN Module was developed and implemented in ns-3 to provide a powerful tool for enabling the simulation of a real LoRaWAN network instead of simulating a simplified MAC protocol. This add-on module allows the research community and developers to more understanding of the behaviour of physical and MAC layers in LoRa networks. Credits to the LoRaSim that allows the users to test a network with varying SFs based on gateway feedbacks



Entry count 12
Node Id All
Add Node Id None
Add Node Id None
Add Node Id None
Apply filter
Show Trajectory

	Time	Node Id	X coord	Y coord
1	0	0	34	50.5
2	0	1	67	50.5
3	0	2	17.5	21.9212
4	0	3	5.42116	34
5	0	4	1	50.5
6	0	5	5.42116	67
7	0	6	17.5	79.0788
8	0	7	83.5	21.9212
9	0	8	95.5788	34
10	0	9	100	50.5
11	0	10	95.5788	67
12	0	11	83.5	79.0788

Fig. 2. This is Ns3 NetAnim Environment

. The LoRaWAN ns-3 module may be used to simulate LoRaWAN networks, and it is available for download here. Beyond the models created to represent different components of the network and the integrated helpers used to set them up, the ns-3 LoRaWAN module includes a packet tracker that can be used to monitor network behaviour and analyze its performance. It also includes facilities for storing the network topology in a file to debug and monitor. It also provides many scenarios as an example of simple to complex network use cases. The integrated LoRaWAN module meets the requirement of Class A devices. That means it can simulate the use cases where devices send uplinks and receive downlinks from the server. In comparison to LoRa 's two other available classes (Class b and Class C), the most power-efficient end-devices is this class. To deliver a highly configurable, agile solution, the physical layer, MAC layer, and transport and use are built. Concerning the LoRaWAN-based ns-3 module's primary features, they are installation of the network server, ADR, confirmed messages and support for multi-GW. Configurability of the LoRaWAN ns-3 module proposed and allows new algorithms to be implemented on the server-side in [21]. Also, the investigational evaluation of LoRaWAN conducted by ns-3 in [22], Improving LoRa performance with CSMA [23], the power consumption model [24], scalability analysis model for significant-scale [19] and several of models in [25-29].

3.3 FloRa

Framework for LoRa (FloRa) simulator was developed to evaluate LoRa networks performance using ADR mechanism. It proved the effectiveness of ADR in increasing the PDR with improving energy efficiency. FLoRa is an end-to-end simulation framework for LoRa networks rely on the OMNeT++ network simulator and also utilizes INET system components [30]. An open-source OMNeT++ library was designed to support the experimentation process for various network protocols. FloRa code is created by C++, and it enables the development of LoRa networks that support the integration of LoRa nodes, gateways, and network server modules. Application logic can be implemented as separate modules that are linked to a network server. The network server and nodes support dynamic configuration parameters controlling via the ADR and considering collisions and capture effect. The module includes an accurate modelling of the backhaul network and can simulate multiple gateways. At the end of the simulation, energy consumption statistics can be collected in each node and over the entire network [31]. Besides, the modules of the LoRaWAN MAC protocol strive to simulate the physical layer [32]. This offers a very strong graphical interface as opposed to the other simulation applications since it is based on OMNeT++ and a graphical network description. The developed simulation module includes a sample scenario in the FLoRa simulations directory. The scenario has several features to simulate a network with ten nodes that are placed randomly in a square network topology with one gateway that is linked to a network server. Each node transmits a packet at a time based on an exponential distribution with a defined mean. For simulating a LoRa network, several parameters need to be selected, such as simulation time and warm-up period, SF, the transmission power for each LoRa end-node, backhaul network configuration,

``

and links. The simulation statistics and tracing files are generated on completion of the run. The simulation statistics can be viewed through the OMNeT++ GUI as in Fig 3.

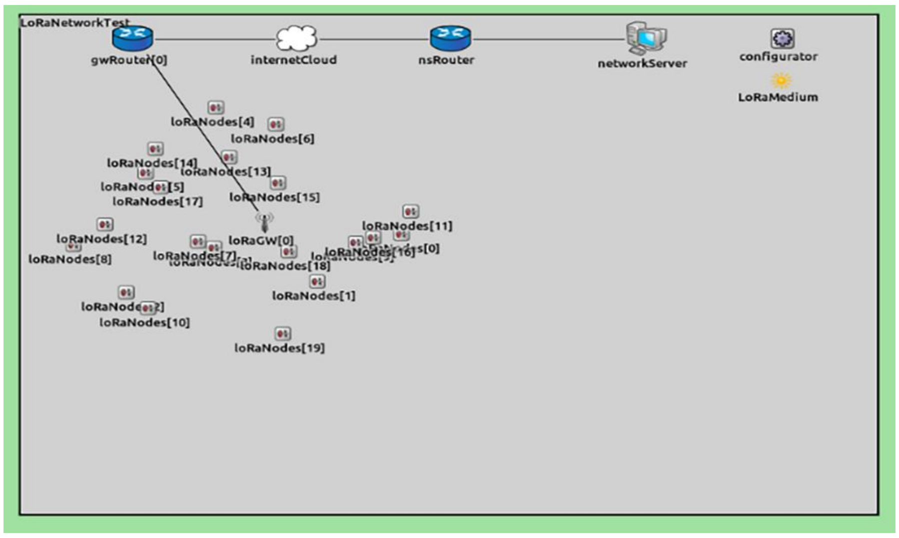
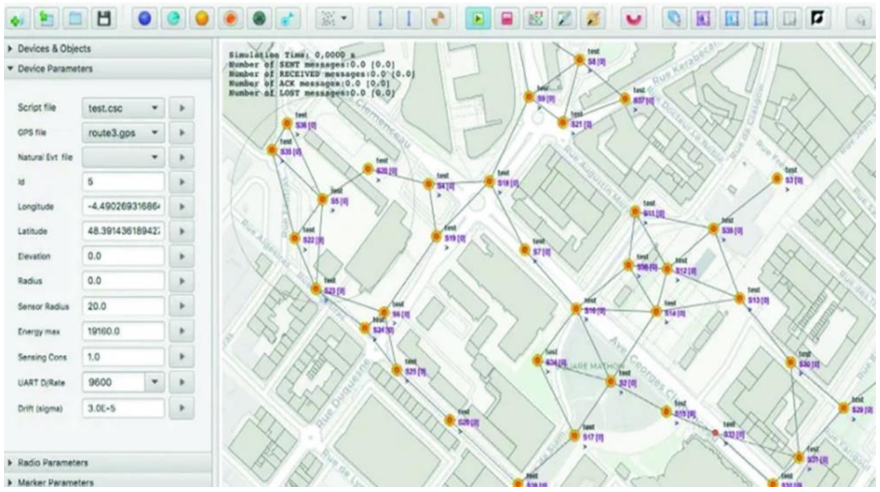


Fig. 3. FLoRa Environment [30]

3.4 CupCarbon

Carbon is an emerging framework for simulating Smart City and IoT WSNs (SCI-WSN) [33]. It aims to design, visualise, debug, and validate distributed algorithms for environment observing and data gathering. It is used to simulate various IoT application scenarios for educational and development scientific projects. In addition, to visually explain the basic concepts of WSNs, it also supports the testing of different wireless topologies, protocols, and applications. CupCarbon supports two simulation environments that enable the design of mobility scenarios (fires and gas scenarios, vehicles and UAVs, insects) and a discrete event simulation of WSNs and IoT applications. Networks can be simulated via an ergonomic and user-friendly GUI utilizing the OpenStreetMap framework to deploy sensors clearly on the map. It has a script named SenScript that enables the programming and configuration of sensor nodes individually and generates codes to be used in hardware platforms like Arduino, Raspberry Pi, and XBee. The nodes can be configured dynamically to distribute nodes into individual networks.

CupCarbon supports the calculation of energy consumption and displays it as a function of the simulation time. Such functionality permits clarification of network structure and realistic implementation before real deployments. It integrates propagation visibility with interference models and supports different communication interfaces, including, LoRa, Wi-Fi and ZigBee protocols [34]. CupCarbon is the fundamental kernel of the PERSEPTEUR ANR project aimed at developing algorithms for the accurate simulation of signal transmission in a 3D city area. [35]. The ground elevation model can be imported into a CupCarbon project by Google Elevation API [36]. Many recent studies were utilized CupCarbon to analyze LoRa/LoRaWAN performance [34, 37, 38]. CupCarbon offers several objects which are easy to use and easy to customize [39]. CupCarbon offers a multi-agent simulation environment that enables simulations to be conducted and events and adjustments over time to be tracked and allows the reproduction of a 3D environment consist of a floor, buildings, and different objects such as sensor nodes as illustrated in Fig 4.



Script file	test.csc
GPS file	route3.gps
Natural Evt file	
Id	5
Longitude	-4.49026931686
Latitude	48.39143618942
Elevation	0.0
Radius	0.0
Sensor Radius	20.0
Energy max	19160.0
Sensing Cons	1.0
UART D/Rate	9600
Drift (sigma)	3.0E-5

```
Simulation Time: 0,0000 s
Number of SENT messages:0.0 [0.0]
Number of RECEIVED messages:0.0 [0.0]
Number of ACK messages:0.0 [0.0]
Number of LOST messages:0.0 [0.0]
```

Fig. 4. CupCarbon Environment

3.5 PhySimulator

PhySimulator was used as a link level assessment for LoRa, which shows that, although theoretical consideration is that spreading factors can be considered orthogonal, LoRa has a real problem with inter-spread factor collisions [5]. The objective of the PHY Simulator is to enforce LoRa's relation level. MATLAB writes PhySimulator. The purpose of the simulator shall test the reception of two LoRa transmissions interfering with various diffraction variables [40]. In particular, each spread factor output is influenced by the packet, symbol, and bit error rate that interfered with some other spread factor. The user can use this simulator to edit several parameters (change the values of the variables codes). For instance, bandwidth, payload, and maximum tests can be modified per phase, etc. Unable to alter all these factors via a graphical interface, the user must edit them directly by changing the MATLAB code. Field test and Capacity simulation of LoRaWAN in a seaport area conducted in [41], An Effective Algorithm for

Vehicular Ad-Hoc Network Load Balancing in [42], smart city [43], Realistic Network Planning [44], latest update by H.Mroue called it LoRa+ [45].

3.6 SimpleIoTSimulator

Many popular IoT protocols are supported by SimpleIoTSimulator like (MQTT, MQTT-SN, MQTT Broker, HTTP/s client/server, Modbus Over TCP, BACnet/IP server, LoRa device, LoRa gateway). LoRa Device simulator part supports the Class A, B and C devices specifications and communicates produced LoRa frames (PHY Payload) over UDP to one or more associated LoRa Gateways. It promotes both Over-the-Air Activation (OTAA) and Over-the-Air Activation (ABP) for security key configuration, as illustrated in Fig 5. When talking to certain gateways, signal strength, signal-to-noise, and channels can be specified[46]. The loading of data can be adjusted to display various sensor behaviour. LoRa Gateway is sent to a designated LoRa Network Server to receive LoRa frames (PHYPayload) from LoRa devices via UDP, encapsulated into Semtech UDP Packet Forwarder (Gateway Message Protocol-GWMP) or Semtech Basic Station (WebSockets) packets. Upon receipt of Network Server messages, downstream the frame payload is extracted and forwarded over UDP to the corresponding device.

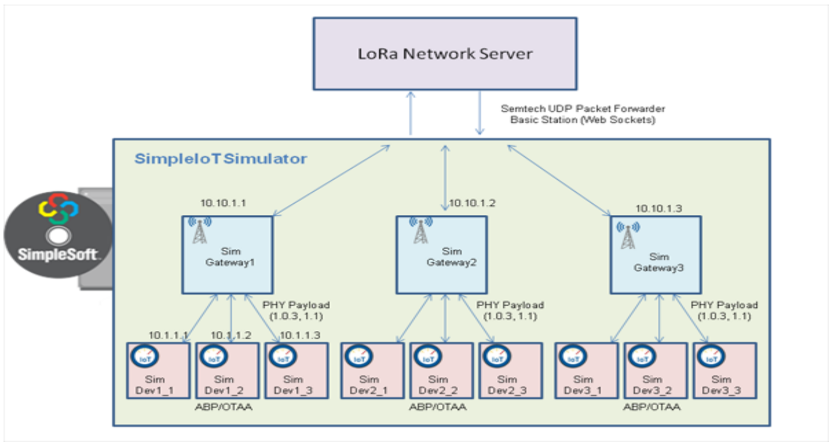
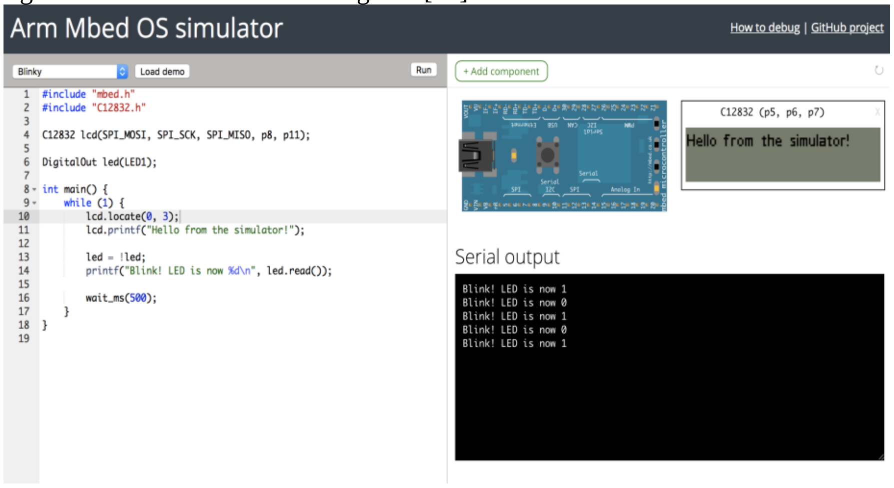


Fig. 5. SimpleIoTSimulator Environment

3.7 Arm Mbed OS Simulator

The simulator is available in two versions: an online version that runs entirely within the browser and an offline version that can be used with any Mbed OS 5 project, as illustrated in Fig 6. The online Mbed Simulator is the simplest. Running the simulator offline on any Mbed OS 5 project is also possible[47]. This makes it possible to incorporate the simulator into your development process. LoRa alliance supports this simulator. The simulator employs a bogus LoRaWAN radio driver in order to run the LoRaWAN stack. When the LoRaWAN stack tries to drive the radio, the counterfeit radio intercepts the packet. This is low enough to encrypt only the packets, data rate, and frequency. This data is then sent directly to a LoRaWAN network server that isn't actually a LoRaWAN device. This method performs well because bidirectional data, recognitions and OTAA all come together[47].



```
#include "mbed.h"
#include "C12832.h"

C12832 lcd(SPI_MOSI, SPI_SCK, SPI_MISO, p8, p11);

DigitalOut led(LED1);

int main() {
    while (1) {
        lcd.locate(0, 3);
        lcd.printf("Hello from the simulator!");

        led = !led;
        printf("Blink! LED is now %d\n", led.read());

        wait_ms(500);
    }
}
```

```
Blink! LED is now 1
Blink! LED is now 0
Blink! LED is now 1
Blink! LED is now 0
Blink! LED is now 1
```

Fig. 6. Arm Mbed OS Simulator

Table 1. LORA SIMULATORS COMPARISON.

Features	**LoRaSim**	**NS-3**	**Flora**	**CupCarbon**	**SimpleIoTSimulator**	**Mbed Simulator**	**PHY Simulator**
License Type	Source is open	Source is open	Source is open	Free (education)	Not Free	Free	Free
Operating System	macOS, Linux, Windows	Linux, Windows,	Linux, Windows, macOS	macOS	Linux: RedHat, Ubuntu, CentOS	Windows.Mac os, Linux.	MacOS, Windows
Installation requirements	SimPy, NumPy, matplotlib	Import all library online	OMNeT++ 6 and INET 4.3.1	Java	n/a	Online, MbedCLI. Python 2.7,Git.	Matlab
Type Language	Python	C++, Python	C++	Java	C++	Java	MATLAB
GUI	Only plot	yes	yes	2D/3D with OSM	yes	yes	Only plot
Community Support	Limited	Very Good	Limited	Good	limited	LoRa Allaince	Good
Last Update	2020	October 2020	Nov 2020	2020	2019	May 2019	2020
Last Version	n/a	ns-3.32	6.0	3.8	n/a	n/a	n/a
Popularly	high	high	medium	little	rarely	high	high
Studies achieved by simulators	[10, 12, 14, 48-56]	[19, 21-29, 57, 58]	[30, 32]	[34, 36-39]	n/a	n/a	[41-45, 59-62]

4 Discussion

We compare the features of the considered simulators in this section to come out with a useful conclusion about the preferences of each simulator. The operating system support, the type of license, interface, the availability of energy usage statistics, and the simulator programming language or environment, latest version, and update are among the feature's comparison in Table I.

All simulators are discrete events and support the LoRaWAN protocol and can model the network as a cycle of discrete events in the time domain. This authorizes the

``

simulators to switch to the next event if two consecutive events do not change, so the system does not need to be monitored continuously. Regarding the languages of programming used in Simulators' implementation, all simulators are designed on well-known, supportive community programming environments. This is very important as a specialist can quickly extend the capability of the simulator with the implementation of new modules as extensions for the existing simulators (e.g., enhance new network protocols, incorporation of further tools in the current environment).

In brief, Java is used for CupCarbon implementation; C++ is used for FloRa; Matlab is used for PhySimulator, Python is used for LoRaSim, and the C++ and Python implementation are used for the Ns-3 LoRaWAN module. In contrast to other simulators, CupCarbon via 2D and 3D environment, FLoRa via OMNeT++, and Ns-3 via NetAnim and PyViz have a broader graphical GUI for C++ and Python module, respectively. Whereas Only a few plots offer PhySimulator and LoRaSim, MATLAB, and Pychrom environments in that order. All of the simulators studied were published in the scientific community, and according to the official websites of each simulator: CupCurbon, FloRa, and PhySimulator have more than two related publications, but PhySimulator has been updated by another version named LoRa+. Whilst Module ns-3 includes more than 20 related publications, and LoRaSim has 11 related publications according to the SCOPUS database, the last update was in Nov 2020, but some simulator has many extended versions with different names, not included in these statistics. The ns-3 is nevertheless an open-source project with a large, large community supporting it [18]. The module ns-3 and LoRaSim have more publications in comparison to PhySimulator, CupCurbon, and FloRa. Therefore, some of the simulators provide more detailed details on the installation process and the use of the equipment on their websites. All of them are open source, and GitHub provides their codes. Whiles SimpleIoTSimulator [46]and Arm Mbed Simulator[47] are more technical using but haven't enough publication projects in spite of the Arm Mbed simulator being adopted by LoRa alliance.

5 Conclusions

In the networking and communication environment, simulation has proved an important tool to evaluate new solutions. However, the simulator must provide an environment similar enough to the re-created environment to produce realistic results and evaluate those solutions appropriately. There are multiple simulators on LoRaWAN, but each has different features that can be tailored to specific assessments. The paper provides a conceptual review of seven LoRaWAN simulators, LoRaSim, NS3, Cupcurpon, FloRa, SimpleIoTSimulator and Mbed simulator. This paper provides a conceptual assessment. The only open-source emulation tools and the most popular in the academic area are LoRaSim, NS3, the Physimulator compared to the other three simulation tools, which are more techniques used. Unlike the basic implementation, the simulators provided in this work add new features to the MAC sublayer that are not available in the basic version. They have been published and are written in a variety of programming languages. To achieve this, there is no need to construct complex real-world networks or sophisticated testbeds. The results of a study that included a review of the literature and research using current simulators revealed that open-source programming environments are capable of modelling the vast majority of MAC layer mechanisms in a LoRaWAN network.

Acknowledgement

This study was supported by the Universiti Malaysia Pahang (www.ump.edu.my), Malaysia, under the Post Graduate Research Scheme PGRS200340.

References

1. Mikhaylov, K., J. Petäjäjärv, and T. Hänninen, *Analysis of the Capacity and Scalability of the LoRa Wide Area Network Technology*. 2016.
2. Attia, T., et al. *Experimental Characterization of LoRaWAN Link Quality*. in *2019 IEEE Global Communications Conference (GLOBECOM)*. 2019.
3. Salama, M., Y. Elkhatib, and G. Blair, *IoTNetSim: A Modelling and Simulation Platform for End-to-End IoT Services and Networking*. 2019. 251-261.
4. Piechowiak, M. and P. Zwierzykowski, *Simulations of the MAC Layer in the LoRaWAN Networks.* Journal of Telecommunications and Information Technology, 2020.
5. Hornbuckle, C.A., *Fractional-N synthesized chirp generator*. 2010, Google Patents.
6. Semtech, *SX1272/3/6/7/8: LoRa Modem Designer's Guide , Available online : www.semtech.com.* 2013.
7. Bouguera, T., et al., *Energy Consumption Model for Sensor Nodes Based on LoRa and LoRaWAN.* Sensors, 2018. **18**(7).
8. Augustin, A., et al., *A Study of LoRa: Long Range & Low Power Networks for the Internet of Things.* Sensors, 2016. **16**: p. 1466.
9. Raza, U., P. Kulkarni, and M. Sooriyabandara, *Low Power Wide Area Networks: An Overview.* IEEE Communications Surveys & Tutorials, 2017.
10. Bor, M., et al., *Do LoRa Low-Power Wide-Area Networks Scale?* 2016.
11. Farooq, M.O. and D. Pesch. *Poster: Extended LoRaSim to Simulate Multiple IoT Applications in a LoRaWAN*. in *EWSN*. 2018.
12. Farooq, M. and D. Pesch, *Extending LoRaSim to Simulate Multiple IoT Applications in a LoRaWAN*. 2018.
13. Zorbas, D., et al., *Optimal Data Collection Time in LoRa Networks—A Time-Slotted Approach.* Sensors, 2021. **21**(4): p. 1193.
14. Abdelfadeel, K., T. Farrell, and D. Pesch, *How to make Firmware Updates over LoRaWAN Possible*. 2020.
15. Zorbas, D., P. Kotzanikolaou, and D. Pesch, *TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things.* Computer Communications, 2020. **153**: p. 1-10.
16. Abdelfadeel, K.Q., et al., *FREE—Fine-Grained Scheduling for Reliable and Energy-Efficient Data Collection in LoRaWAN.* IEEE Internet of Things Journal, 2020. **7**(1): p. 669-683.

``

17. Henderson, T.R., et al. *ns-3 project goals*. in *Proceeding from the 2006 workshop on ns-2: the IP network simulator*. 2006.
18. Simulator, N., *: https://www.nsnam.org/ access date Nov 2020.*
19. Abeele, F.V.d., et al., *Scalability Analysis of Large-Scale LoRaWAN Networks in ns-3.* IEEE Internet of Things Journal, 2017. **4**(6): p. 2186-2198.
20. Bilalb, S.M. and M. Othmana, *A performance comparison of network simulators for wireless networks.* arXiv preprint arXiv:1307.4129, 2013.
21. Reynders, B., Q. Wang, and S. Pollin, *A LoRaWAN module for ns-3: Implementation and Evaluation* in *Proceedings of the 10th Workshop on ns-3 - WNS3 '18*. 2018. p. 61-68.
22. Khan, F.H. and M. Portmann, *Experimental Evaluation of LoRaWAN in NS-3.* 2018 28th International Telecommunication Networks and Applications Conference (Itnac), 2018: p. 453-460.
23. To, T. and A. Duda. *Simulation of LoRa in NS-3: Improving LoRa Performance with CSMA*. in *2018 IEEE International Conference on Communications (ICC)*. 2018.
24. Finnegan, J., S. Brown, and R. Farrell, *Modeling the Energy Consumption of LoRaWAN in ns-3 Based on Real World Measurements.* 2018 Global Information Infrastructure and Networking Symposium (Giis), 2018.
25. Stellin, M., S. Sabino, and A. Grilo, *LoRaWAN Networking in Mobile Scenarios Using a WiFi Mesh of UAV Gateways.* Electronics, 2020. **9**(4).
26. Hariprasad, S. and T. Deepa, *Improving Unwavering Quality and Adaptability Analysis of LoRaWAN.* Procedia Computer Science, 2020. **171**: p. 2334-2342.
27. Finnegan, J., R. Farrell, and S. Brown, *Analysis and Enhancement of the LoRaWAN Adaptive Data Rate Scheme.* Ieee Internet of Things Journal, 2020. **7**(8): p. 7171-7180.
28. Khan, F.H., R. Jurdak, and M. Portmann, *A Model for Reliable Uplink Transmissions in LoRaWAN.* 2019 15th International Conference on Distributed Computing in Sensor Systems (Dcoss), 2019: p. 147-156.
29. Finnegan, J., S. Brown, and R. Farrell, *Evaluating the Scalability of LoRaWAN Gateways for Class B Communication in ns-3.* 2018 Ieee Conference on Standards for Communications and Networking (Ieee Cscn), 2018.
30. Mariusz Slabicki, G.P., Mario Di Francesco, *https://omnetpp.org/download-items/FLoRA.html Accessed:2020.* 2017.
31. site., F.o., *FLoRa.* 2020.
32. Mariusz Slabicki, G.P., Mario Di Francesco, *Adaptive Configuration of LoRa Networks for Dense IoT Deployments by OMNet++(FLORA) simulator.* 2018.
33. Cupcarbon., *Cupcarbon. http://cupcarbon.com. Accessed:2020.*
34. Lopez-Pavon, C., S. Sendra, and J.F. Valenzuela-Valdés, *Evaluation of CupCarbon Network Simulator for Wireless Sensor Networks.* Netw. Protoc. Algorithms, 2018. **10**(2): p. 1-27.
35. Cupcarbon. *Cupcarbon User Guide*. 2020; Available from: https://cupcarbon.com/cupcarbon_ug.html.
36. Lounis, M., et al., *3D Environment for IoT Simulation Under CupCarbon Platform*. 2017.
37. Sanchez, E.B. and D.F.H. Sadok. *LoRa and LoRaWAN Protocol Analysis Using Cupcarbon*. in *International Congress of Telematics and Computing*. 2020. Springer.

38. John, A., et al., *Energy Management and Monitoring Using IoT with CupCarbon Platform*, in *Green Computing in Smart Cities: Simulation and Techniques*. 2020, Springer. p. 189-206.

39. Mehdi, K., et al., *CupCarbon: A Multi-Agent and Discrete Event Wireless Sensor Network Design and Simulation Tool.* 2014.

40. D. Croce, M.G., S. Mangione, G. Santaromita and I. Tinnirello, *website: http://lora.tti.unipa.it/.* 2018.

41. Bardram, A.V.T., et al., *LoRaWan Capacity Simulation and Field Test in a Harbour Environment.* 2018 Third International Conference on Fog and Mobile Edge Computing (Fmec), 2018: p. 193-198.

42. Abbas, A.H., L. Audah, and N.A.M. Alduais. *An Efficient Load Balance Algorithm for Vehicular Ad-Hoc Network*. in *2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS)*. 2018.

43. Yu, F.H., Z.M. Zhu, and Z. Fan, *Study on the feasibility of LoRaWAN for smart city applications*. 2017 Ieee 13th International Conference on Wireless and Mobile Computing, Networking and Communications (Wimob), 2017: p. 334-340.

44. Dalela, P.K., S. Sachdev, and V. Tyagi, *LoRaWAN Network Capacity for Practical Network Planning in India.* 2019 Ursi Asia-Pacific Radio Science Conference (Ap-Rasc), 2019.

45. Mroue, H., et al., *LoRa+: An extension of LoRaWAN protocol to reduce infrastructure costs by improving the Quality of Service.* Internet of Things, 2020. **9**: p. 100176.

46. SimpleSoft. *SimpleIoTSimulator*. 2019 [cited 2021 23/06/2021]; Available from: https://www.simplesoft.com/SimpleIoTSimulator.html.

47. Lab, M. *Mbed Simulator*. 2019 [cited 2021 23/06/2021]; Available from: https://os.mbed.com/blog/entry/introducing-mbed-simulator/.

48. Farooq, M.O. and D. Pesch. *Evaluation of Multi-Gateway LoRaWAN with Different Data Traffic Models*. in *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*. 2018.

49. Hassan, K., *Resource Management and IP Interoperability for Low Power Wide Area Networks*. 2020.

50. Abdelfadeel, K., V. Cionca, and D. Pesch, *A Fair Adaptive Data Rate Algorithm for LoRaWAN*. 2018.

51. Bor, M. and U. Roedig. *LoRa Transmission Parameter Selection*. in *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. 2017.

52. Voigt, T., et al., *Mitigating Inter-network Interference in LoRa Networks.* 2016.

53. Bor, M., A. King, and U. Roedig, *Lifetime Bounds of Wi-Fi Enabled Sensor Nodes.* 2015. **52**: p. 1108-1113.

54. Bor, M. and U. Roedig, *OpenCL as Wireless Sensor Network Programming Abstraction*. 2014.

55. Ta, D.-T., et al., *LoRa-MAB: A Flexible Simulator for Decentralized Learning Resource Allocation in IoT Networks*. 2019. 55-62.

56. Ta, D., et al. *LoRa-MAB: Toward an Intelligent Resource Allocation Approach for LoRaWAN*. in *2019 IEEE Global Communications Conference (GLOBECOM)*. 2019.

``

57. Reynders, B., et al., *Improving Reliability and Scalability of LoRaWANs Through Lightweight Scheduling.* IEEE Internet of Things Journal, 2018. **5**(3): p. 1830-1842.
58. Magrin, D., M. Centenaro, and L. Vangelista. *Performance evaluation of LoRa networks in a smart city scenario thesis*. in *book*. 2017.
59. Tomic, I., et al., *The Limits of LoRaWAN in Event-Triggered Wireless Networked Control Systems.* 2018 Ukacc 12th International Conference on Control (Control), 2018: p. 101-106.
60. Croce, D., et al., *Impact of LoRa Imperfect Orthogonality: Analysis of Link-Level Performance. matlab.* IEEE Communications Letters, 2018. **22**(4): p. 796-799.
61. Gucciardo, M., I. Tinnirello, and D. Garlisi. *Demo: A Cell-level Traffic Generator for LoRa Networks. matlab*. 2017. ACM.
62. Marini, R., W. Cerroni, and C. Buratti, *A Novel Collision-Aware Adaptive Data Rate Algorithm for LoRaWAN Networks.* IEEE Internet of Things Journal, 2021. **8**(4): p. 2670-2680.