Efficient Decoding of Synchronized Colliding LoRa Signals

Samira Abboud, Nancy El Rachkidy, Alexandre Guitton
Université Clermont Auvergne, CNRS, LIMOS, F-63000 Clermont-Ferrand, France
samira.abboud@isima.fr, nancy.el_rachkidy@uca.fr, alexandre.guitton@uca.fr

***Abstract*—In LoRa (Long Range), when a collision occurs in the network, each end-device has to retransmit its colliding frame. This reduces the throughput, and increases the energy consumption of the end-devices and the delay of the frames. In this paper, we propose an algorithm to decode colliding synchronized LoRa signals and thus improve the overall performance of the network. Indeed, we use successive transmissions of bitmaps by the end-devices to determine the correct symbols of each colliding frame, instead of retransmitting the whole frames. Simulation results show that our algorithm is able to significantly improve the overall throughput of LoRaWAN, and to decrease the energy consumption and the delay of the transmitters.**

keywords - LoRa, LoRaWAN, collision cancellation, synchronized signals.

I. Introduction

Internet of Things (IoT) installations are becoming a reality and networks are being deployed to achieve smart cities, such as transportation, healthcare, surveillance systems and environmental monitoring applications [1]. For example, the climate conditions, such as fires, earthquake, flooding, and volcanoes are monitored, where sensors may send data to the server over non-saturated network. Many of these IoT installations rely on Low-Power Wide-Area Network (LPWAN) technologies. These emerging technologies such as Long Range (LoRa) [2], Sigfox [3], RPMA [4] and Weightless [5] enable power-efficient wireless communication over very long distances.

LoRa [2] is a widely deployed LPWAN technology and is considered by a large number of industries as a base for their IoT applications. LoRa uses orthogonal transmission settings (such as frequency and spreading factor) to reduce collisions. However, collisions cannot be totally avoided even when considering the capture effect. Current LoRa deployments use a default behaviour for retransmissions of colliding frames where each transmitter (e.g. the end-device) has to retransmit the whole colliding frame, which leads to reduce the throughput, and to increase the energy consumption and the delay. For these reasons, it is desirable to find a method for decoding the colliding signals, while decreasing the delay and the energy consumption, and improving the throughput.

In this paper, we consider non-saturated traffic conditions where the performance of both LoRaWAN and our proposed algorithm are analyzed in terms of the actual observed throughput, delay and energy consumption. Besides, we consider the case of fully synchronized signals. We show that overlapped symbols can be extracted by the receiver, although it is not possible to determine to which frame each symbol belongs. We propose an algorithm which has the benefit of decoding such superposed signals while reducing the impact of collisions. The proposed algorithm relies on end-devices sending bitmaps in order to determine the correct symbols of the frames, instead of retransmitting the whole frames. This algorithm always succeeds to retrieve the frames from the superposed signals regardless of the number of colliding transmitters. This is done by retransmitting bitmaps until the receiver has been able to decode everything (i.e until all frames are successfully decoded), so the process only stops when everything is decoded. Furthermore, the delay, energy consumption, and throughput are improved compared to LoRaWAN. our proposed algorithm.

The rest of the paper is as follows. Section II presents a description of LoRa and LoRaWAN and some related works. Section III describes our proposed algorithm used for decoding fully synchronized LoRa signals. Section IV presents our simulation results. Finally, Sect. V concludes this paper.

II. State of the art

In the following, we first describe the physical layer LoRa and the medium access control (MAC) protocol LoRaWAN. Then we present some of the related works.

A. The LoRa physical layer

Long Range (LoRa) is a proprietary Chirp Spread Spectrum modulation technique by Semtech [2]. The Spreading Factor (SF) is the main LoRa parameter. SF has an influence on the transmission duration, the energy consumption, and the communication range. SF defines the number of bits encoded into each symbol, and varies between 7 and 12. Each symbol encodes one of 2^{SF} values and covers the entire frequency band. LoRaWAN [6] is an open protocol which defines a MAC layer for LPWAN technology, and operates on top of LoRa. LoRaWAN architecture is composed of end-devices that are connected to the network server through gateways which relay messages. LoRaWAN enables three classes of operation for end-devices: class A, class B, and class C. In class A, end-devices send data when ready and wait for an acknowledgment from the network server. Then, they switch to sleep mode to save energy until the next transmission. The delay between two transmissions has to be larger than 99 times the duration of the frame transmission in order to respect the duty cycle of 1%. In class B, which is optional, end-devices have additional scheduled receive periods to allow downlink communications with a bounded delay. In class C, which is also optional, end-devices are always active.

B. Related work on synchronized signals on LoRa

Few works [7], [8] and [9] have been done to study the collisions among synchronized signals in LoRa.

In [7], the authors worked on constructing an efficient multi-hop network based on the sub-GHz LPWAN technology. They investigated the combination of LoRa and concurrent transmission (CT) which is a flooding protocol that considers synchronized packet collisions that happen when multiple relays perform immediate retransmissions at the same time. They found that, due to the time domain and frequency domain energy spreading effect, LoRa is robust to the packet collisions resulting from CT. They found the receiver performance under CT can be further improved by introducing timing offsets between the relaying packets. Therefore, they proposed a timing delay method, the offset-CT method, that adds random timing delay before the packets while preventing the timing offset from diverging over the multi-hop network. Their experiments demonstrate the improvement brought by the CT method. Moreover, they showed that LoRa survives the CT purely by capture effect. In this paper, we decode LoRa colliding signals without considering the capture effect.

In [8], the authors presented Choir which is a system that improves throughput and range of LPWANs. Choir proposed a novel approach that exploits the natural hardware offsets between low-cost nodes to separate collisions from several LPWAN transmitters using a single-antenna base station. Choir improves the throughput by decoding transmissions from multiple nodes simultaneously. Specifically, signals from two transmitters are likely to experience a small frequency offset, due to a difference in the frequency of their oscillators. This results in the two chirps being slightly offset in frequency.

In [9], the authors proposed two algorithms to decode some cases of collisions of LoRa signals. The first algorithm is used when superposed signals are slightly desynchronized, and the second algorithm when superposed signals are completely synchronized. Authors observe that the first algorithm is able to significantly increase the throughput, by decoding many collisions of two signals, and some collisions of three signals. On the other side, the second algorithm has improved the throughput, by decoding both signals when exactly two signals are colliding. The second algorithm requests any of the two colliding transmitters to retransmit its frame. Hence, when one frame is retransmitted, the algorithm is able to decode it, and to deduce the other colliding frame by elimination. The authors have considered the case of only two synchronized signals.

III. Propositions

In this section, we present our contributions for the physical and MAC layers. We propose an algorithm to decode superposed signals that are fully synchronized. These signals are received on the same channel, with the same SF, and with the same received signal power. The proposed algorithm can not be applied on LoRaWAN protocol since most communications in LoRaWAN are desynchronized. Therefore, we introduce a new MAC layer which could be used on top of LoRa.

A. Problem statement for the physical layer

In this subsection, we consider the superposition of signals from transmitters (i.e. end-devices) that are fully synchronized as in [9]. Recall that papers [7] and [8] have ensured the feasibility of the synchronization among LoRa signals, by implementing a real LoRa system where transmissions were synchronized in time. Figure 1 shows an example of the reception of two fully synchronized signals under SF7. The figure shows that the signals start at the same time. Frame transmitted by end-device 1 (referred to as ED1) is $f_1 = (64, 32, 32)$, and frame transmitted by ED2 is $f_2 = (96, 0, 32)$. In this figure, we can see that the receiver can extract symbols $\{64, 96\}$ during the first symbol duration, symbols $\{32, 0\}$ during the second symbol duration, and symbol $\{32\}$ during the third symbol duration. However, the receiver is not able to determine to which frame each symbol belongs.

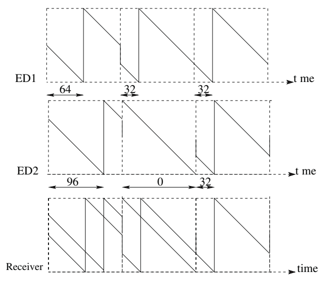

Fig. 1: Superposition of two synchronized signals in uplink.

B. Our MAC protocol

In this subsection, we present and explain our MAC protocol used to decode synchronized colliding signals.

1) Description of MAC protocol and timing computation: Figure 2 shows a MAC protocol implementing our proposed algorithm depicted for four end-devices. Beacons are sent by the gateway to synchronize the communications. We assume that signals are received at the gateway with similar power. In some cases, power control at the transmitter can be used to enforce this hypothesis. Otherwise, the capture effect might hold, and only the frame with the strongest signal would be decoded. We assume that the time is divided into slots and that transmissions on the same slot are fully synchronized. When a collision occurs between frames, the gateway stores all superposed symbols at each symbol duration. Then, it sends a frame built from these symbols. The gateway waits for bitmaps from the end-devices in order to decode the colliding frames, where each bit corresponds to the symbol chosen in the gateway frame. The gateway frame contains, in addition to the symbols, the order of the EDs using an identifier on one symbol. As long as the frames sent by the EDs are not yet decoded, the gateway sends a new frame, and the EDs reply by sending new bitmaps. If there is a collision between a frame and a bitmap, both are lost, and the Gw retransmits a bitmap later.

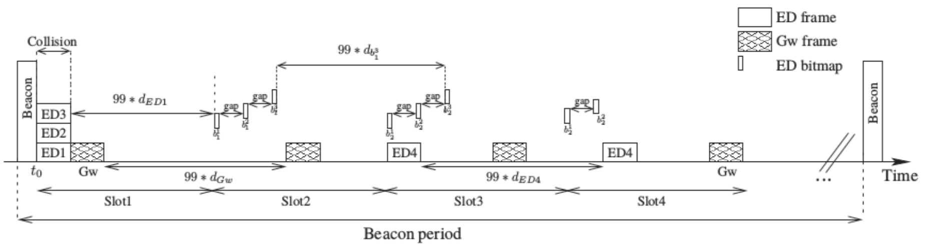

Fig. 2: New MAC protocol depicted for the superposition of four synchronized signals.

To further describe our new proposed MAC protocol, we develop a timing computation model for the transmission process. Specifically, we consider separately the first and the subsequent transmission attempts.

The first frames transmission: The first transmission attempt of the end-devices frames is made on Slot1 in our example, where three end-devices ED1, ED2 and ED3 are initially fully synchronized, and sent their uplink frames at the same initial start time t_0. This causes a collision at the gateway Gw which stores the superposed symbols, and sends a frame composed from these superposed symbols.

The bitmaps first transmission: Each bitmap of a given end-device is separated from the bitmap of the previous and the next end-devices by a small amount of time called guard interval (i.e gap). This guard time ensures that the bitmap of an end-device does not collide with the bitmap of another end-device, even when considering clock drifts.
In Fig. 2, each ED in Slot2 sends a bitmap in reply to the Gw frame. The bitmaps b_1^1, b_1^2, and b_1^3 are sent by ED1, ED2 and ED3 respectively, after 99 times the duration of the frame transmission of ED1, ED2 and ED3. In addition, for each end-device x, its bitmap b_i^x is delayed to avoid a collision with b_i^{x-1} and b_i^{x+1}. In other words, for an ED x, the start time $t_{b_i^x}$ of its bitmap number i should respect the following rule: $(t_{b_i^{x-1}} + d_{b_i^{x-1}}) \times (1 + \Delta_{max}) \leq (t_{b_i^x}) \times (1 - \Delta_{max})$ where $d_{b_i^{x-1}}$ is the time on air of the previous bitmap, and Δ_{max} is the maximum drift. Hence, the start time $t_{b_i^x}$ of a bitmap b_i^x is given by the following equation: $t_{b_i^x} = t_{b_i^{x-1}} + d_{b_i^{x-1}} + gap$. Moreover, the duration of a slot is given by the following: $d_{Slot} = \max(d_{ED} + d_{Gw}, d_{b_i^x} \times x + gap \times (x-1) + d_{Gw})$ where d_{ED} is the duration of the end-device frame, and d_{Gw} is the duration of the gateway frame.
In relation to the slot duration, we set the start time of the last slot as follows: $t_{Slot_{nmaxslots}} = d_{Slot} \times (nmaxslots - 1)$ where $nmaxslots$ is the maximum number of slots.

The bitmaps subsequent transmissions: The start time of a bitmap for the subsequent tranmissions is given by $t_{b_i^x} = \max(t_{b_{i-1}^x} + 100 \times d_{b_{i-1}^x}, t_{Gw} + 100 \times d_{Gw})$ where the ED should respect the duty cycle regulation, and should wait for the Gw frame before sending its bitmap b_i^x with $i > 1$.

Specific case: The general MAC algorithm contains slots where collisions between bitmaps and the frames of end-devices may occur. As shown in Slot3 of Fig. 2, the frame of ED4 collides with the bitmaps b_2^1 and b_2^2 related to ED1 and ED2 respectively. The gateway receives the bitmap b_2^3 of ED3 and decodes it, while bitmaps b_2^1 and b_2^2 are not received. This leads ED1 and ED2 to retransmit their colliding bitmaps in Slot4. Also, ED4 retransmits its colliding frame as shown in Slot4 while respecting the duty cycle of 1%. In addition, the frame of a given end-device may collide with all the bitmaps of the other y end-devices that are sent on the same slot if: $d_{ED} \geq y \times d_{b_i^y} + (y-1) \times gap$

2) Description of our algorithm: Our algorithm described by Algorithm 1 is used to decode fully synchronized colliding signals for X transmitters (i.e end-devices), with $X \geq 2$.
When a collision occurs for X EDs, the gateway can not decode the colliding frames, and is not able to determine to which frame each symbol belongs. Hence, the Gw considers that all the end-devices frames contain missing symbols, represented by * in Algorithm 1. Meantime, the gateway sends a frame built from the superposed symbols. At this step, each ED x replies to the gateway frame by sending a bitmap b_i^x (which corresponds to the bitmap number i). For each bit at position j in b_i^x, the algorithm checks the value of the bit. If bit is equal to 1, then the algorithm replaces * in the frame of ED x with the current symbol j of the gateway frame. On the other hand, if bit is equal to 0, the algorithm checks if the number of superposed symbols at the current position j is equal to 2. If it is the case, then the algorithm replaces the * with the other current symbol at position j of the superposed symbols. In addition, the algorithm verifies if at position j, all the symbols of the EDs have been decoded (i.e not equal to *), and if there is still a missing symbol (i.e *) in a frame of another ED y. If it is the case, then the algorithm replaces the * in the symbol j of y by the remaining current symbol in the same position j of the superposed symbols. As long as there are missing symbols that can not be decoded by the gateway, the process is repeated until the decoding of all colliding signals. It is worth mentioning that this algorithm runs in polynomial time.

C. Guessing the frame

In this subsection, we show that the choice of symbols by the gateway has an impact on the number of bitmaps transmissions needed for each end-device. We choose $f_1 = (64, 32, 32)$, $f_2 = (96, 0, 32)$, and $f_3 = (96, 64, 32)$ to explain our proposed algorithm which is described by Algorithm 1.

Step 1: The gateway sends a frame with the following arbitrary set of symbols $f_{G1} = (64, 0, 32)$. ED1 replies with the bitmap $b_1^1 = (1, 0, 1)$, ED2 replies with $b_1^2 = (0, 1, 1)$, and ED3 with $b_1^3 = (0, 0, 1)$. The current data frame of ED1 corresponds to $f_1 = (64, *, 32)$, the current data frame of ED2

Algorithm 1: Decoding of fully synchronized superposed signals.

```
while a frame contains $*$ do
    the gateway sends a frame
    for each end-device $x$ do
        $x$ sends a bitmap $b_i^x$
        for each bit at position $j$ in $b_i^x$ do
            if $bit = 1$ then
                symbol $j$ of the frame $f_x \longleftarrow$ symbol $j$ of the gateway frame
            if $bit = 0$ and the number of superposed symbols at the current position $j$ is $2$ then
                symbol $j$ of $f_x \longleftarrow$ the other symbol at position $j$ of the superposed symbols
        for each end-device $y$ do
            for each symbol at position $j$ do
                if $y \neq x$ and all symbols $j$ of $f_x \neq *$ and the symbol $j$ of $f_y = *$ then
                    symbol $j$ of $f_y \leftarrow$ the remaining symbol $j$ of the superposed symbols
```

corresponds to $f_2 = (96, 0, 32)$, and the current data frame of ED3 corresponds to $f_3 = (96, *, 32)$.

Step 2: Since some of the frames of the EDs still contain missing symbols, the Gw sends another frame $f_{G2} = (96, 0, 32)$. ED1 replies with $b_2^1 = (0, 0, 1)$, and ED3 with $b_2^3 = (1, 0, 1)$. The updated frames of ED1 and ED3 remain the same as in *Step 1*. ED2 did not reply since its frame was decoded in *Step 1*.

Step 3: Since some of the frames of the EDs still contain missing symbols, the Gw sends another frame $f_{G3} = (96, 32, 32)$. ED1 replies with $b_3^1 = (0, 1, 1)$, and ED3 with $b_3^3 = (1, 0, 1)$. Hence, f_1 is $(64, 32, 32)$, and f_3 is $(96, 64, 32)$. Note that the second symbol of f_3 is decoded by deduction. In this example, the average number of transmissions for each ED is 2.33 bitmap transmissions.

Another example to decode the aforementioned colliding frames is the following:

Step $1'$: same as *Step 1*

Step $2'$: The Gw sends $f'_{G2} = (96, 32, 32)$. Therefore b_2^1 is $(0, 1, 1)$, and b_2^3 is $(1, 0, 1)$. Hence f_1 is $(64, 32, 32)$, and f_3 is $(96, 64, 32)$. Here, the average number of bitmap transmissions for each end-device is 1.66 transmissions.

Therefore, the choice of the symbols by the gateway may impact the number of needed bitmap transmissions for each end-device. Hence, we propose a random selection of symbols that are not already sent by the gateway.

D. Theoretical number of bitmap transmissions

In this subsection, we evaluate theoretically the average number of bitmap transmissions per end-device needed to decode the collided frames. We consider a collision between three frames f_1, f_2 and f_3 for ED1, ED2 and ED3 respectively, with each frame composed from three symbols.

1) Best case scenario: $f_1 = f_2 = f_3 = (96, 64, 32)$. In this example, there is no need to send any bitmap because the receiver is able to determine the symbols of each frame since all symbols are equal in each symbol duration.

2) Worst case scenario: $f_1 = (96, 64, 32)$, $f_2 = (64, 96, 0)$ and $f_3 = (0, 32, 96)$. In this example, all symbols are different from each other in each symbol duration. The average number of transmissions for each end-device is 2 bitmap transmissions.

We deduce that the average number of bitmap transmissions per ED needed to decode its collided frame varies between 0 and $(X-1)$, where X is the number of end-devices in collision.

IV. Simulations and Results

In this section, we study and evaluate the performance of our proposed algorithms in terms of delay of successful decoding of colliding signals, energy consumption, and throughput.

A. Parameter settings

Simulations were done using our own simulator developed in Java. We model a network with a single Gw, and X EDs. We assume that all EDs transmit with the same SF and on the same channel, and that their signals are received with almost the same strength at the Gw, i.e., no capture effect occurs. The error detection subject is not investigated. The Gw duty cycle remains a bottleneck when more EDs are present. Hence we assumed a few/very low-rate EDs in the range (since LBT is not used). The duty cycle determines the frequency of re/transmissions. It guarantees fairness between all EDs. Hence, it is important that EDs/Gw obey this cycle time since LBT is not used by LoRaWAN. Here, the duty cycle of 1% is respected. We use both SF7 and SF12, as these two SFs have the highest and lowest data rates, respectively. The frame length is set to 30 bytes. We assume that the network is not saturated [1]. We used $nmaxslots = 1000$, the maximum number of EDs $X = 8$, and $gap = 30\ ns$. Simulation results are obtained by averaging over one thousand samples.

B. Numerical number of bitmap transmissions in our protocol vs Numerical number of frame retransmissions in LoRaWAN

The maximum number of retransmissions in LoRaWAN is set to 8 attempts by default [6]. Hence, after 8 retransmissions attempts in LoRaWAN, a loss might arise for the colliding frames. We observed for instance, that for 8 colliding EDs, we have almost 7.5 frame retransmissions per ED with 90.93% of frame losses in LoRaWAN, while we have only 6.5 bitmap transmissions per ED without frame losses in our algorithm.

C. Average delay

Figure 3 shows the average evolution of the delay for the correct decoding of a frame in LoRaWAN and in our algorithm. The delay specifies the difference between the full decoding time of the frame by the Gw and the first

[1]We chose the non-saturated case in order to better analyze the network performance. The saturated case will be handled in an extension of this paper.

time it is sent by the ED. We notice that the delay in LoRaWAN is greater than in our algorithm. This is due to the transmission of short bitmaps in our algorithm, which have shorter size compared to a complete frame in LoRaWAN, which leads to reduce the transmission time. For example, for 4 colliding EDs, we observe a decrease in the delay of 30% with SF12, and of 20% with SF7 in our algorithm compared to LoRaWAN.

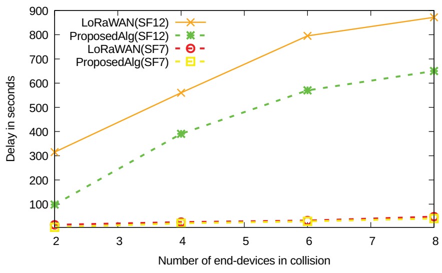

Fig. 3: Average delay of superposed signals.

D. Average energy consumption

Figure 4 shows the average energy consumption per useful bit calculated for an ED after the full decoding of its frame by the Gw in both LoRaWAN and our algorithm. It is observed that the consumed energy increases with the increase of the number of colliding signals, and also with the increase of SF. We refer to the following equation to compute the consumed energy per useful bit: $E_{bit} = (\frac{P_{cons}(P_{tr})*(N_{Payload}+N_P+4.25)*2^{SF}}{8*PL*BW})$ [10], where $P_{cons}(P_{tr})$ is the total consumed power that depends on transmission power. We set P_{tr} to 14 dBm [6]. PL is the payload size set to 30 bytes. $N_{Payload}$ is the number of symbols used to transmit the payload and it is set to 30 for SF12 and 45 for SF7. N_P is the preamble length set to 10. BW is the bandwidth set to 125000 Hz. We found that the energy consumption in LoRaWAN is greater than in our algorithm. This is due to the transmission of bitmaps in our algorithm instead of the transmission of whole frames in LoRaWAN, which leads to decrease the energy of the colliding frame. We observe a decrease of 65% with SF7, and 78% with SF12 in the energy consumption in our algorithm in comparison with LoRaWAN.

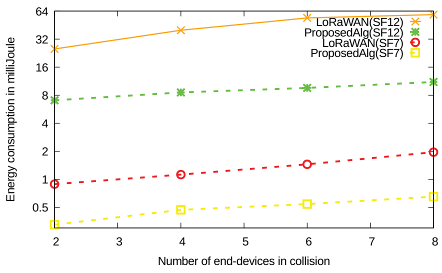

Fig. 4: Average energy consumption of superposed signals.

E. Average throughput

Figure 5 shows the average evolution of the throughput computed for an ED at the Gw side in both LoRaWAN and our algorithm. The throughput is already small in LoRaWAN, and it decreases further with the number of EDs and collisions. In Fig. 5, we observe that the throughput for an ED in LoRaWAN is smaller than that in our algorithm. This is due to the delay in LoRaWAN which is greater than the delay in our algorithm, and which leads to further decrease LoRaWAN throughput. In addition, we have frame losses in LoRaWAN, which increase with the number of EDs. This increase leads to decrease further the throughput in LoRaWAN compared to our algorithm where no frame losses are present. For instance, for 8 colliding EDs, we observe a gain of almost 95% with SF12, and 27% with SF7 in our algorithm compared to LoRaWAN.

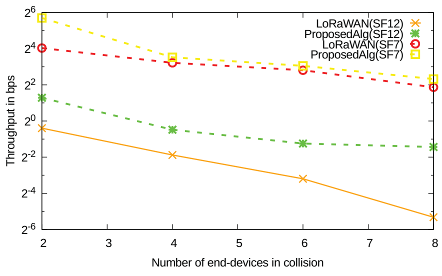

Fig. 5: Average throughput after a collision.

V. Conclusion

Collision is a factor that negatively impacts LoRaWAN throughput, which is already very limited (between 250 and 11000 bps). In this paper, we propose a collision resolution algorithm that enables to decode colliding frames in LoRa. Our algorithm focuses on the case where end-devices are fully synchronized. The proposed algorithm relies on retransmitting bitmaps in reply to guesses from the gateway instead of the whole frame. Based on our simulation results, we show that the proposed algorithm is able to improve the throughput, by decoding the frames in collision. This algorithm is also able to reduce the energy consumption of the end-devices, and to decrease the delay needed to decode the frames. These results contributed to the development of a new MAC protocol based on LoRaWAN, relying on the proposed collision resolution algorithm, and surpassing LoRaWAN.

References

[1] S. Talari, M. Shafie-khah, P. Siano, V. Loia, A. Tommasetti, and J. P. Catalão, "A review of smart cities based on the Internet of Things concept," *Energies*, vol. 10, no. 4, p. 421, 2017.
[2] Semtech, "AN1200.22 LoRa^{TM} Modulation Basics." https://www.semtech.com/uploads/documents/an1200.22.pdf.
[3] Sigfox. http://www.sigfox.com.
[4] IngenuRPMA. http://www.ingenu.com/technology/rpma/.
[5] W. O. Standard. http://www.weightless.org. Accessed:2015-11-07.
[6] L. Alliance, "LoRaWAN specification," *LoRa Alliance*, 2015.
[7] C.-H. Liao, G. Zhu, D. Kuwabara, M. Suzuki, and H. Morikawa, "Multi-hop LoRa networks enabled by concurrent transmission," *IEEE Access*, vol. 5, pp. 21430–21446, 2017.
[8] R. Eletreby, D. Zhang, S. Kumar, and O. Yağan, "Empowering Low-Power Wide Area Networks in Urban Settings," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 309–321, ACM, 2017.
[9] N. El Rachkidy, A. Guitton, and M. Kaneko, "Decoding Superposed LoRa Signals," *IEEE LCN*, 2018.
[10] T. Bouguera, J. Diouris, J. Chaillout, R. Jaouadi, and G. Andrieux, "Energy Consumption Model for Sensor Nodes Based on LoRa and LoRaWAN.," *Sensors (Basel, Switzerland)*, vol. 18, no. 7, 2018.