

Article

Edge-Based and Gateway-Based SmartSync Systems for Efficient LoRaWAN

Mohammad Al mojamed 

Computing Department, Engineering and Computing College–Al-Qunfudhah, UMM AL-QURA University, Makkah 24382, Saudi Arabia; mmmojamed@uqu.edu.sa

Abstract

Low-Power Wide-Area Networks (LPWANs) like LoRaWAN enable IoT applications with low-power and long-range characteristics. While LoRaWAN class B mode is server-initiated downlink communication-oriented, its uplink communication, especially in mobile scenarios, remains underexplored. This paper proposes two novel systems, Edge-based SmartSync and Gateway-based SmartSync, aiming to enhance uplink by leveraging class B synchronization. Edge-based SmartSync enables end devices to dynamically adjust the Spreading Factor (SF) based on real-time Received Signal Strength Indicator (RSSI) from beacons, achieving a significant improvement in terms of packet delivery and energy consumption. Gateway-based SmartSync ensures the fair distribution of end devices across a lower SF to further enhance the efficiency of the system. The beacon is reengineered to convey sensitivity limits to end devices. The systems were implemented in the OMNeT++ simulator over a 25 km² area with 100–1000 mobile devices and evaluated against a baseline using metrics like the Packet Delivery Ratio, collisions, and energy consumption. The obtained results show that both systems are capable of improving the delivery ratio by over 40% and reducing collisions by 80% compared to the baseline, with energy savings exceeding 35%. Proposed systems offer cost-effective, adaptable solutions, paving the way for more reliable IoT deployments.

Keywords: class B; LoRa; LoRaWAN; LPWAN; SF allocation

1. Introduction

LPWAN (Low-Power Wide-Area Network) technologies have been used as a fundamental network infrastructure for several Internet of Things applications over the past years. LPWAN technology is designed to connect a massive number of end devices over long distances, with low data rates, while keeping power consumption at a minimum level. It provides an alternative solution for standard cellular communication infrastructure. It enables a broad range of IoT applications in diverse environments due to its scalability and efficiency. LPWAN facilitates innovations across different sectors, including environmental monitoring, healthcare, smart cities, industrial automation, agriculture, and transportation.

Several LPWAN technologies have emerged to fill the gap in the required infrastructure for IoT applications. Sigfox, NB-IoT, and LoRaWAN are the most popular technologies among the different available LPWAN technologies [1,2]. These technologies differ in terms of communication range, data rate, and scalability. For instance, LoRaWAN typically supports communication ranges of up to 10–15 km in suburban environments, with data rates ranging from 0.3 kbps to 50 kbps, while Sigfox offers ultra-low data rates (up to 100 bps) with very limited payload sizes. NB-IoT, on the other hand, provides higher data



Academic Editor: Reza K. Amineh

Received: 3 March 2026

Revised: 27 March 2026

Accepted: 28 March 2026

Published: 30 March 2026

Copyright: © 2026 by the author.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

rates and reliability but at the cost of increased energy consumption and reliance on cellular infrastructure. Among these technologies, LoRaWAN is particularly attractive due to its flexibility, low deployment cost, and ability to support large numbers of devices.

Sigfox is a proprietary technology that provides wide global coverage, low cost, and low power solutions for IoT applications with over 20 million interconnected devices. It is based on Free License bands offering long-distance communication with low data rates [3]. NB-IoT, on the other hand, is another LPWAN technology based on the 3GPP standard to provide narrowband radio technology capable of offering deep coverage and long battery life. NB-IoT operates within existing cellular infrastructure, where it leverages its security infrastructure to provide robust communication for IoT applications [4,5].

Moreover, LoRaWAN [6] has emerged as a promising solution for IoT applications that require long-distance transmission with low energy consumption. LoRaWAN is characterized by its ability to deliver data of up to 10–15 km range in suburban areas [7]. LoRaWAN can also save up to 10 times the power compared to NB-IoT when transmitting the same payload over a similar period [8]. It has been approved by the International Telecommunication Union (ITU) as the standard for LPWAN technologies [9]. LoRaWAN is a star topology that allows a large number of end devices to communicate only with the gateways, but there is no direct communication between end devices. Gateways, thereafter, take responsibility for forwarding edge device traffic toward the cloud network server. The IP networks, including Ethernet, Wi-Fi, cellular networks, and satellites, can be used to connect gateways to the cloud. LoRaWAN operates as the data link layer over the physical LoRa, which is a radio frequency modulation created by Semtech.

From the perspective of devices, LoRaWAN offers three distinct classes: class A, class B, and class C [10]. Class A must be implemented by every LoRaWAN end device, whereas the other two classes are extensions to the main class A and can be optionally implemented by devices based on the requirements of the developed IoT application. Uplink and downlink communications are supported by all three classes. In class A, uplink and downlink communication are initiated by end devices. A device can send an uplink message at its will and must be followed by two receive windows that can be used for downlink communication. However, class B provides support for applications that require server-initiated downlink communication. This is achieved through implementing synchronization using periodically broadcast beacons to determine when to open a periodic receive window by an end device. It allows the network server to know when to send a downlink message to an end device. Class C, on the other hand, extends the functionality of class A by maintaining the receive window open all the time except when transmitting. This makes devices available to receive downlink traffic at any time. However, this comes at a cost of consuming more power.

LoRaWAN is primarily used for uplink-centered applications, such as telemetry. However, LoRaWAN use cases have expanded to downlink-centered applications, such as actuators and emergency systems, which require periodic downlink with a balance between power consumption and end-to-end delay. Therefore, end devices are not only traffic initiators but also receive orders, updates, and instructions from the main platform. These downlink-centered applications are based on the synchronization provided by class B mode. The use of class B operation presents the opportunity for other uplink-based applications to enhance their performance, especially in network scenarios that involve end device mobility, i.e., devices are mobile where they change their location frequently.

Furthermore, in LoRaWAN class B networks, where periodic beacon synchronization is used to enable scheduled communication, most existing studies focus on improving synchronization accuracy or reducing downlink latency, with limited attention given to uplink performance optimization. In particular, there is a lack of lightweight mecha-

nisms that dynamically adapt SF allocation in mobile scenarios while maintaining compatibility with standard LoRaWAN operation. This gap motivates the need for efficient and scalable solutions that can improve uplink reliability without introducing significant protocol modifications.

The work in this paper makes a significant contribution to the field of LoRaWAN technology by proposing two systems, namely, Edge-based SmartSync and Gateway-based SmartSync. They leverage existing class B Mode features to enhance communication efficiency in IoT applications. Edge-based SmartSync enhances performance by enabling end devices to dynamically select the most suitable Spreading Factor (SF) based on real-time channel quality that can be retrieved from beacon messages. Complementing this, a Gateway-based SmartSync is proposed to further improve network efficiency. Gateway-based SmartSync ensures the fair distribution of end devices across lower SFs. It involves reengineering beacons to convey information regarding balanced device distribution across lower SFs to all the end devices, reducing collisions and demonstrating scalability. These systems offer cost-effective, adaptable solutions that address the limitations of traditional systems, paving the way for more reliable IoT deployments in diverse environments.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 gives an overview of LoRaWAN class B and introduces the proposed systems. The evaluation methodology is presented in Section 4. The result of the evaluation is then presented and discussed in Section 5. Finally, the conclusion of the paper, along with the future directions, is presented in Section 6.

2. Related Works

A study was presented in [11] to identify the performance and scalability issues in the LoRaWAN class B operation environment. It proposed an algorithm capable of reconfiguring class B scheduling and synchronization parameters to reach optimal operation in either cases, network sizes, or application requirement changes. An Adaptive Beacon Period Configurator (ABPC) is proposed to set the value of several class B parameters based on the changes that may occur in network size or traffic load. The system mainly targets adjusting the main class B beacon period (from the standard 128 s) to better fit the number of end devices as the network grows. The beacon guard and beacon reserved are untouched and are set with their standard values. However, major changes are introduced in the beacon period, and hence, the beacon window. The beacon size, the periodicity, and the number of ping slots and their duration are configured whenever there is a change in the network size. The number of end devices is used as a trigger for the proposed system. It assumes cases with a 1024 s beacon period and about one thousand end devices.

The work in [12] focused on analyzing the performance of classes B and C, as it claimed that most of the research is on class A, and fewer studies were dedicated to classes B and C. The study investigated the effect of several network parameters on the overall performance of LoRaWAN. It studied the impact of using parallel reception paths at gateways, the use of retransmission techniques in confirmed traffic mode, and the effect of preferring transmission over reception in class B and C end devices. The paper observed that increasing the number of parallel reception paths affects performance positively. The use of prioritizing reception over transmission was found to have no impact on the performance of networks with unconfirmed traffic, whereas it affected the overall performance negatively in the case of confirmed traffic network applications. It also suggested the optimal number of transmissions to be eight or fewer for classes B and C, where more than this number would lead to unfavorable results.

Another work that analyzed LoRaWAN performance is [13]. However, this study is based on real-world LoRaWAN traffic that was monitored in four European cities. The

study has found several issues of LoRaWAN deployment, one of which is the misalignment of LoRaWAN class B beacon messages. Several causes were identified for such misalignments. One of them is the faulty implementation of converting between the GPS time and the Coordinated Universal Time (UTC), which resulted in an 18 s offset. Another reason for the misalignment is the existence of some malfunctioning gateways with an incorrect system clock that leads to random network problems. Moreover, the difference between UNIX time and GPS was also found to be another reason for the misalignment, where gateways were found using the UNIX time system and broadcasting beacons to GPS-based devices. The study then suggested the use of class B beacons as a timebase source for synchronization in urban networks instead of using the Global Navigation Satellite System (GNSS). The class B beacons were favored as they do not require a clear sky view compared to the GNSS, and hence they are more suitable for scenarios with buildings or other interfering structures.

A class B beacon was utilized in [14] to enhance the performance of LoRaWAN uplink communication in a network operating with class B mode. It proposed an uplink scheduling scheme to avoid collisions between uplink frames. The scheme is mainly based on the frequently received beacons from the gateway. Each end device calculates its uplink scheduling slots using a proposed pseudorandom generator (PRS). The proposed pseudorandom generator makes sure to calculate a unique offset for each device to avoid two devices transmitting simultaneously. In order to do that, the pseudorandom generator combines three factors to calculate the seed number. In addition to the device's address and beacon time, which are used to compute class B beacon offset, the PRS uses the channel frequency as well to determine the uplink slots within each channel frequency for each device. The procedure is repeated whenever a device receives a new beacon from the gateway. Despite all that, collisions may still occur due to possible overlapping of determined uplink slots by different devices. Moreover, the difference in airtime for an uplink, especially those transmitted using high SFs, can still pose possible collisions.

The adaptive data rate (ADR) mechanism has been optimized in [15] to enhance the downlink communication in LoRaWAN operating class B mode. However, some modifications were introduced to the ADR mechanism to make it more suitable for class B mode environments. To avoid affecting the ping slot timing of the class B beacon, gateways were restricted to changing the data rate up to DR3 only. Another key point of ADR that was dismissed is the power increment before changing the data rate, since gateways are battery-less and operate on the maximum permitted power. Gateways then keep track of statistics for each device, and whenever it is time to send downlink traffic to a specific device, they will use the computed data rate based on the collected information. Moreover, the required statistics will not be available to the gateways unless there is some uplink traffic from the end devices to the gateways. Therefore, the proposed system suggests collecting such information from any uplink traffic if there is any or to be piggybacked to downlink confirmation messages, which are sent from the device upon receiving a downlink traffic. Gateways also maintain the volume of lost downlink traffic with each device. If they reach a predetermined threshold, the end device is notified of a newly decreased data rate that will be used on the following downlink communication.

The work in [16] investigated the performance of LoRaWAN class B in terms of downlink throughput, downlink delay, and energy consumption using the M/G/1 queueing model. It considers frame waiting time at the gateway as well as the power consumption of the desired end device. A cost function was proposed to optimize a beacon parameter known as parameter k . The cost function is proposed to reach one as the waiting time and power consumption decrease. The cost function is then used to figure out the key parameter, hence determining the optimal number of ping slots within a beacon frame. A

direct relation between the number of optimal ping slots and the density of downlink traffic was found. Another trade-off relation was noted between end device power consumption and frame waiting time at the gateway. It is justified by the fact that a smaller k value results in a low number of ping slots but with long periods, causing a higher waiting time. On the contrary, this led to less power consumption by end devices.

Changes in the structure of the uplink frame were introduced in [17] to enhance LoRaWAN class A and class B efficiency. Standard uplink frame in LoRaWAN is usually followed by two receiving windows, RX1 and RX2, that are open after predefined delays known as the Rx1 delay and the Rx2 delay. The two windows are used to transmit any downlink traffic from the network server to end devices, such as MAC commands and configuration parameters. However, the work proposed to re-order the slots of the uplink frame in class B, where end devices should open the RX2 window in advance of transmitting an uplink message. A device starts by sending a ping message to a gateway to express its readiness for transmitting an uplink packet. RX2 is then opened to receive configuration parameters from the gateway. The uplink transmission then takes place. The RX1 window is opened following the transmission, similar to standard LoRaWAN. The goal of such modifications is to convey the best configuration parameters to the end device prior to uplink transmission. This includes the SF, channel, and data rate.

The work in [18] made an effort to synchronize uplink transmission by introducing a new class based on class B. The same timeline as class B is used in the proposed mechanism to synchronize uplink communication instead of downlink communication. End devices can calculate when to send their uplink messages. The ping slots, in the standard beacon, are redefined as transmission opportunities. However, the basic division of the beacon window in standard class B, 4095 slots, is not used. Instead, a new division mechanism was proposed. The window is virtually split into several intervals to accommodate uplink transmission, provided that each interval is wide enough for an uplink. The number of intervals, number of transmission opportunities, is a configured parameter. Another considered parameter when calculating the number of intervals is the time between two transmission opportunities. All other features of the proposal are similar to those in standard class B, including beacon reserved, offset, beacon guard, and pseudorandom calculation.

The class B synchronization mechanism was also exploited by [19] to provide for uplink synchronization. A time slot, an SF, and a channel are scheduled for each uplink transmission to avoid any uplink transmission collision and enhance the network reliability. The proposed system consists of two phases. In phase one, the pre-configuration phase, the network server is assumed to know the location of end devices, and it allocates the SF and the group index for each participating device. The network server divides the area into six circles, where each of them is allocated an SF. The group index is determined by the network server based on the distance of an end device from the gateway, as well as the number of devices in the same group. Each group is limited to four devices using the same SF. However, a group index may have devices from different annuli and be allocated different SFs. The second phase involves allocating a time slot and a channel for uplink communication. This is done via introducing a new beacon message that is broadcast after each regular class B beacon. The window of the newly added beacon is divided into several uplink slots based on the total number of group indices. However, this would undoubtedly double the amount of standard class B beacons in the network.

The work in [20] has addressed integrated LoRaWAN with an LEO satellite. It proposed a beacon-based uplink LoRaWAN (BU-LoRaWAN) to improve satellite–terrestrial network performance. LoRaWAN class B synchronization was utilized to provide effective uplink communication from ground-based LoRaWAN devices to orbiting satellite gateways. A pull-based traffic transmission mechanism was used as an opportunistic

networking technique based on class B mode. The beacon window within the beacon period is identified as an opportunity frame for uplink transmission. This is justified by the fact that the reception of beacons means the satellite gateways are reachable and currently flying over the current end devices. Thus, it is a good opportunity for the end device to transmit its uplink traffic. The proposal also uses a queue data structure to enqueue end devices' ready-to-send traffic until the appropriate transmission time. The queue is implemented at the AMC layer within each device and is frequently monitored for any uplink traffic. Possible transmission collision is avoided using a random transmission slot within the beacon window.

LoRaWAN class B was used in [21] for real use case applications. The paper proposed the use of class B end devices to control smart grid operation and the coordination of the distributed Interference Protection System in smart and micro-grids. It investigates the scalability of such use to find the maximum number of end devices that can be served by a single gateway, providing maximum response time to fit the targeted application, the distributed Interference Protection System. The response time is the time between an event triggered by an end device and the reception of the related coordination command. The study found that a maximum of 312 Interference Protection Systems can be handled by a single gateway, while the maximum response time that complies with the Italian regularity framework is theoretically respected.

The authors in [22] proposed LoRaSync as a class B-based synchronization scheme to provide time-slotted uplink transmission in LoRaWAN [23]. The proposed system targeted minimizing power consumption using the beacon skipping technique. The work measured standard LoRa clock drift to find how fast drifting away from time reference occurs in LoRa devices. The results are then used in LoRaSync to construct large enough slots to avoid clock skews. The maximum clock drift and the constructed slot length are then used to find the number of beacons that can be skipped to decrease power consumption while maintaining time synchronization. Hence, the skipping approach would contribute to the overall power saving and extend battery life. However, LoRaSync was found to reduce the overall throughput because of the reduced number of slots resulting from stretching slot length. A methodology was then introduced to determine the best slot size based on the required traffic, maintaining a balance between throughput and power consumption.

The performance of class B mode has been evaluated and analyzed in several works in the literature using network simulators [24,25]. The first focused on investigating the delay and the delivery efficiency for downlink traffic using LoRaWAN class B compared to class A. A significant improvement was found for class B over class A in terms of link access delay and delivery efficiency for downlink traffic mode. The second studied the scalability of class B and found that the duty cycle is the main reason behind the deterioration in performance in a large networks operating in class B. Another work that comprehensively analyzes class B operation is [26], focusing on the downlink communication in terms of transmission efficiency, delay, and power consumption. It evaluated the impact of tuning class B configurable parameters to identify the optimal configuration that suits actuator-based applications. The authors in [27] have explored the usability of a time-slotted medium access technique for LoRaWAN instead of Aloha. The paper identified duty cycle limitation, different slot lengths, scheduling, and time synchronization as the main challenges that hinder the adaptability of the time-slotted medium access protocol.

Several studies have investigated techniques to improve uplink performance in LoRaWAN class B networks, particularly in dense deployments where packet collisions and inefficient SF allocation can significantly degrade reliability. Many existing approaches focus on improving synchronization mechanisms between end devices and gateways through enhancements to beacon operation or timing alignment. These methods aim to improve the

stability of class B synchronization and ensure reliable beacon reception. However, most of these approaches mainly address synchronization accuracy rather than optimizing uplink transmission parameters, such as SF allocation.

Other works attempt to improve uplink efficiency through modified channel access mechanisms, scheduling strategies, or gateway-assisted optimization techniques. More broadly, existing approaches can be categorized into three groups: synchronization enhancement methods, channel access optimization schemes, and scheduling-based approaches. Synchronization-focused methods aim to improve beacon reliability and timing alignment in class B networks, while channel access and scheduling techniques primarily target collision reduction and scalability in dense deployments. However, these approaches either do not address SF allocation or require significant modifications to the LoRaWAN MAC layer. In addition, most existing works assume static device deployments and do not explicitly consider mobility. In contrast, the proposed SmartSync systems leverage existing class B beacon synchronization together with RSSI-based SF distribution to dynamically balance SF assignments among devices. This lightweight approach improves uplink reliability while maintaining compatibility with standard LoRaWAN operation and supporting mobile IoT devices.

Table 1 provides a comparative summary of representative LoRaWAN class B uplink optimization approaches and highlights their key differences with respect to the proposed SmartSync systems.

Table 1. Comparison of representative LoRaWAN class B uplink optimization approaches.

Approach	Main Objective	Mobility Support	Protocol Modification
Adaptive beacon configuration for LoRaWAN class B [11]	Dynamically adjusts beacon parameters to improve synchronization performance	No	Minor modification to beacon operation
Collision reduction access mechanism [15]	Improves channel access strategy to reduce packet collisions in dense LoRaWAN networks	No	MAC layer modification
Synchronization improvement for class B networks [16]	Enhances synchronization accuracy between end devices and the gateway	No	Moderate modification
Gateway-assisted optimization scheme [18]	Uses gateway-side information to improve network performance and reliability	No	Additional coordination required
Scheduled uplink transmission scheme [21]	Allocates time slots to devices to reduce collisions	No	Major MAC changes
SmartSync	Dynamic SF allocation using beacon synchronization and RSSI	yes	Minimal modification

3. SmartSync LoRaWAN

In this section, the class B operation is introduced, and then two proposed systems are presented. The first is an Edge-based SmartSync LoRaWAN system that utilizes available class B synchronization information to enhance uplink communication performance. The second proposed system makes the most use of class B synchronization, where the gateway side is more involved in controlling communication settings to improve performance and address mobility.

3.1. Class B Operation Overview

LoRaWAN end devices, using class B, are synchronized with the network side using beacon signals. The synchronization procedure makes end devices available for downlink communication at a predictable time. The synchronization is achieved using a beacon broadcast that is transmitted by gateways periodically, every 128 s. Based on the synchro-

nized time reference, an end device can open a periodic receive window known as a ping slot. The ping slot can be used by the network server to initiate a downlink communication. Figure 1 shows the timing of the class B beacon signal. A beacon period, 128 s, consists of beacon reserved, beacon window, and beacon guard. A beacon transmission starts at the beginning of the beacon reserved slot. It is then followed by 122.8 s of the beacon window. The beacon window is divided into 4096 ping slots. Each ping slot has its own index number and lasts for 30 ms. Therefore, an end device with a W index number should listen at $(\text{beacon reserved} + W \times 30 \text{ ms})$ following the start time of a beacon (Equation (1)).

$$\text{Listening Time} = \text{beacon reserved} + \text{index number} \times 30 \text{ ms} \quad (1)$$

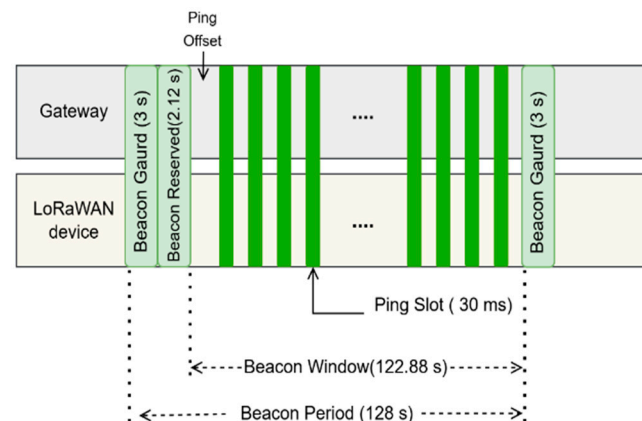


Figure 1. Class B beacon timing.

The number of pin slots for an end device, PingNumber , is controlled by the end device's predefined ping slot periodicity, where it ranges from 0 to 7; 0 means the end device opens a listening slot every one second, whereas 7 means that the device opens the listening slot only once during the beacon window (Equation (2)).

$$\text{PingNumber} = 2^{7-\text{periodicity}} \quad (2)$$

The delay between two ping slots for an end device is known as a ping period and is calculated according to Equation (3).

$$\text{Ping period} = \frac{\text{beacon period}}{\text{number of ping slots}} \quad (3)$$

The last component of a beacon period is the beacon guard, during which ping slots cannot be allocated. The length of the beacon guard represents the time on air for the longest possible frame. It always precedes the following beacon message to make sure that a collision does not occur between the next beacon and the last ping slot of the previous beacon window in case it was used.

Although class B offers synchronization of end devices with the network server side, uplink traffic communication is carried out using class A operation based on the standard. However, synchronization can be of benefit, providing substantial signal quality information, especially in communication scenarios that involve device mobility; i.e., devices changing their locations frequently. Location changes would lead to the case where devices use transmission parameters that no longer hold for the new location. Therefore, periodically transmitted beacons of class B can be utilized to enhance the performance of such scenarios.

3.2. Edge-Based SmartSync LoRaWAN

The proposed system aims to improve LoRaWAN uplink transmission by optimizing the existing class B synchronization mechanism. This mechanism operates solely on the end device side, with no changes required at the gateway or network server. In class B, end devices are synchronized with the network through broadcast messages. A beacon window with multiple slots is allocated for downlink traffic, and beacon messages are broadcast every 128 s to maintain synchronization. In the Edge-based SmartSync LoRaWAN system, end devices optimize these beacon messages to determine the appropriate SF for uplink traffic transmission to gateways.

When an end device receives a beacon message from a gateway, it retrieves the Received Signal Strength Indicator (RSSI) of the signal. In the proposed system, the end device uses this RSSI value to determine the SF for subsequent uplink transmissions. The RSSI reflects the quality of the connection between the gateway and the device, as well as the proximity of the LoRa device to the gateway, which initiates the beacon. A higher RSSI indicates a stronger connection, allowing the use of a lower SF. The device stores the RSSI from the latest gateway communication and uses it to select an appropriate SF for upcoming uplink transmissions. Each SF requires a minimum RSSI for successful signal demodulation, and the end device chooses the SF best suited to the current RSSI. Figure 2 illustrates the pseudo-code for this proposed approach.

```

1  # Initialize stored RSSI
2  STORED_RSSI = 0
3  WHILE TRUE:
4      IF BEACON_RECEIVED THEN
5          # Receive beacon message from gateway
6          beacon = GET_BEACON_MESSAGE()
7          # Extract RSSI from received beacon
8          current_rssi = GET_RSSI(beacon)
9          # Store the latest RSSI value
10         STORED_RSSI = current_rssi
11         get Power_Of_recieved_Signal in Watt
12         mW= Power_Of_recieved_Signal * 1000
13         # Convert to dBmW
14         dBmW = 10.00 * log10 (mW)
15         # Pass the dBmW to LoRawAN app
16         # Find the matching SF based on Reciever Sensitivity
17         IF found_SF != current_SF THEN
18             current_SF = found_SF
19         END IF
20     End IF
21     # Wait for next beacon (every 128 seconds)
22     WAIT_FOR_NEXT_BEACON(128)
23 END WHILE

```

Figure 2. Pseudo-code of Edge-based SmartSync LoRaWAN.

The proposed system adheres to standard LoRaWAN class B specifications without requiring any modifications. As a result, it incurs no additional costs for implementing SF allocation on LoRaWAN end devices and does not involve extra message exchanges. Additionally, the system addresses the needs of mobile LoRaWAN end devices by dynamically assigning the most suitable SF based on the channel quality between the end device and the gateway, accommodating frequent location changes.

3.3. Gateway-Based SmartSync LoRaWAN

The Edge-based SmartSync method effectively assigns communication parameters to end devices. However, it may allocate the same SF to a large number of devices due to their closeness to the gateway, increasing the likelihood of collisions among devices that transmit using a similar SF. Therefore, a Gateway-based SmartSync system is proposed to take such an issue into consideration. The system aims to balance the number of devices on the available SFs, especially the lower SFs. This is achieved by controlling the limit of sensitivity for each single SF through LoRaWAN gateways.

A gateway is used to record the most recent information about end devices when it receives a packet from them. This includes the used SF, the RSSI of the received signal, and the ID of the end device. In addition, it also maintains the overall number of known devices.

The algorithm is triggered when the number of recently known devices reaches a threshold. The threshold can be configured based on different circumstances, such as the network size, or it can be configured based on the capacity of the gateway or traffic density. The threshold is used to trigger the transition from the initial edge-based operation, where devices adapt their SF based on real-time channel conditions, to the gateway-based mechanism, which refines the overall SF distribution as the network grows. This procedure can be done either at the gateway or the network server.

The Gateway-based SmartSync begins by finding the number of devices per SF and sorting the known devices, operating on SF7, SF8, and SF9, by their RSSI values. When the total of devices that use SF7, SF8, and SF9 is greater than half of the known devices by a gateway, the proposed algorithm initiates the process of modifying the lower limit of the minimum signal strength for the previously mentioned SFs to distribute devices fairly among the SFs. The fair distribution refers to achieving a balanced allocation of devices across SF7, SF8, and SF9 based on their RSSI values. The objective of this is to avoid too many devices transmitting with similar SFs and hence increase the chance of collisions. The list of ascendingly sorted RSSIs is divided into three parts. The first value of the first third is assigned as the lower limit of SF9. The RSSI value that exists in the first position of the second third of the list is assigned to SF8's lower limit. Finally, the SF7 lower limit takes the first RSSI value of the third part of the list. This ensures that devices that are capable of transmitting with these SFs are fairly distributed between these three SFs. The decision rule can be formalized mathematically as follows.

Let N denote the number of devices that use SF7, SF8, and SF9, and let $RSSI_i$ represent the received signal strength of device i , where $i = 1, 2, 3, \dots, N$. The gateway collects RSSI values from devices and sorts them in ascending order to form a list. The sorted list is then partitioned into three groups of approximately equal size to determine the SF allocation thresholds. Let $k = \lfloor \frac{N}{3} \rfloor$. The thresholds for SF allocation can be defined as $T_1 = RSSI_{(k)}$, $T_2 = RSSI_{(2k)}$. The SF for each device is assigned according to Equation (4):

$$SF_i = \begin{cases} SF7, & RSSI_i \geq T_2 \\ SF8, & T_1 \leq RSSI_i < T_2 \\ SF9, & RSSI_i < T_1 \end{cases} \quad (4)$$

The algorithm within each cycle produces the lower limits of SF7, SF8, and SF9. The counter of known nodes is re-initialized to zero to begin a new cycle. These newly calculated limits of signal strength for each SF are then broadcast to all end devices to indicate the required changes. They will be reflected in the subsequent uplink messages by end devices.

Note that a LoRaWAN beacon is transmitted in a radio packet that consists of a preamble and a beacon payload BCNPayload. The beacon payload, which contains the beacon

frame, can be 17 or 19 bytes, depending on the regional parameters. The beacon frame format contains RFU fields (Reserved for Future Use) that are utilized by the proposed system while maintaining compatibility with the LoRaWAN specification. Figure 3 shows the modification of the beacon frame format to incorporate the newly added three parameters in the proposed Gateway-based SmartSync.

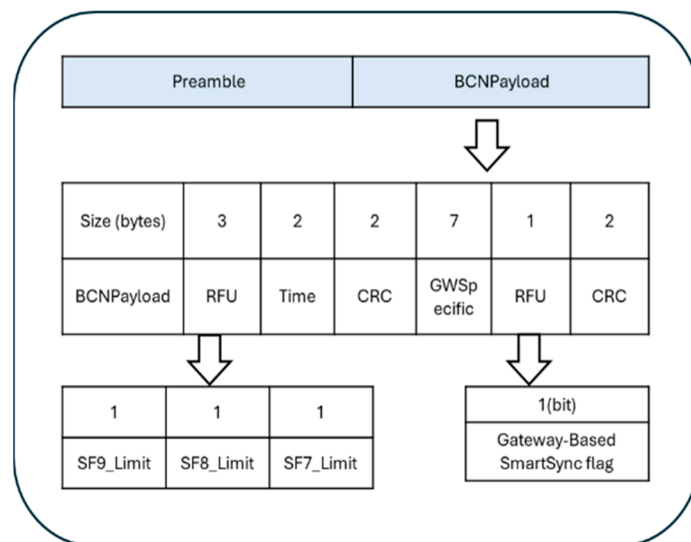


Figure 3. Gateway-based SmartSync beacon frame structure.

The first RFU is a fixed three-byte field. Each byte is used to carry one of the calculated RSSI limit values. However, the intended RSSI limit value is a negative floating-point number and requires efficient encoding to avoid violating beacon payload constraints. Therefore, Equation (5) is used to decode each of the calculated RSSI limits by the proposed system to fit in the RFU. The approximate receiver sensitivity ranges from -136 to -40 according to the standard LoRa radio. Such a value needs to be mapped to a signed 8-bit integer using custom scaling. This gives us a scaling factor of 2.667 ($255/96$), where 255 is the byte capacity and 96 is the range width for -136 and -40 .

$$encodedRSSI = round\left((actualRSSILimit + 136) * \frac{255}{96}\right) \quad (5)$$

The *encoded RSSI* values can then be incorporated into the frame payload in the RFU field contiguously, where *SF7 RSSI limit*, *SF8 RSSI limit*, and *SF9 RSSI limit* are allocated first, second, and third byte, respectively.

Moreover, the Gateway-based SmartSync system uses another bit of the beacon format to indicate to participating end devices that control of the receiver sensitivity is in operation. This would allow end devices to switch from the first method, Edge-based SmartSync, to the Gateway-based SmartSync method. A one bit of the second RFU, which is a one-byte field reserved explicitly as well for future extension, is assigned to controlling receiver sensitivity by the gateway, as shown in the figure.

The presented modification of the beacon frame complies with the LoRaWAN standards. The used RFUs are explicitly designated for any protocol extensions. Despite the used regional parameters, the introduced changes to the beacon frame would work universally. This is because the RFUs are identical across different regions, as the frame structure difference is in the info field only. Therefore, the proposed modification adheres to the LoRaWAN specifications. The pseudo-code of the gateway operations is depicted in Figure 4.

```

1  # Initialize known devices
2  total_known_devices = 0
3  gateway_smart_sync_active = false
4  # Data Structures
5  STRUCT EndDeviceInformation
6      device_id: string
7      sf: integer#Current Spreading Factor
8      rssi: float#Latest RSSI value (dBm)
9  known_devices: map<device_id, EndDeviceInformation>
10 WHILE TRUE:
11   IF MESSAGE_RECEIVED THEN
12     #Extract information from arriving message
13     endDeviceInformation = Extract (arriving message)
14     IF device_id NOT IN known_devices
15       total_known_devices++
16     END IF
17     update known_devices(endDeviceInformation)
18     IF total_known_devices >= MAX_KNOWN_DEVICES_THRESHOLD
19       # count devices using SF7 SF8 SF9
20       devices_using_SF_7_8_9 = count (known_devices)
21       IF devices_using_SF_7_8_9 > (total_known_devices / 2)
22         gateway_smart_sync_active = true
23         rssi_list = Sort_RSSI_SF7_SF8_SF9(known_devices)#Ascending order
24         rssi_limits_SF9 = rssi_list[0]#First third
25         rssi_limits_SF8 = rssi_list[rssi_list.Length() / 3]#Second third
26         rssi_limits_SF7 = rssi_list[(rssi_list.Length() / 3) * 2]#Third third
27         #Encode each RSSI to 8-bit integer using the following
28         (rssi + 136) * (255/96)
29         reset total_known_devices, known_devices
30       END IF
31     END IF
32 END WHILE

```

Figure 4. Gateway pseudo-code for Gateway-based SmartSync LoRaWAN.

The end devices are supposed to operate the first proposed method unless they are instructed via beacon messages that the second strategy is in operation. On receiving a beacon, an end device checks first the bit flag of Gateway-based SmartSync in the second RFU. If the proposed algorithm is on, the end device extracts the RSSI of the received beacon. The extracted RSSI of the signal will work as the reference for the SF that should be used.

The devices then fetch the incorporated SF7 RSSI limit, SF8 RSSI limit, and SF9 RSSI limit from the received beacon. They apply Equation (6) to the received values to recover the original RSSI values. If the signal's RSSI falls within the range of the arrived values, the SF of the current device is changed accordingly. Otherwise, the SF may stay with its old value that was chosen by the first method. This would be the case for devices operating on SF10, SF11, and SF12. The chosen SF will then be used by any subsequent uplink transmission. The pseudo-code of the end device procedure is shown in Figure 5.

$$decodedRSSI = \left(\left(encodedRSSI * \frac{96}{255} \right) - 136 \right) \quad (6)$$

```

1 WHILE TRUE:
2   IF Beacon_RECEIVED THEN
3     # Recieve beacon from gateway
4     beacon=GET_BEACON_MESSAGE()
5     IF beacon.rfu2[0] == 1 # Gateway-Based SmartSync is On
6       set Gate_Based_SmartSync On
7       curren_rssi = ExtractSignalRSSI(beacon)
8       # Extract and decod the recieved limit
9       # using decodedRSSI = ((encodedRSSI*96/255) - 136)
10      sf9_limit= DecodeRSSI(beacon.rfu1[0])
11      sf8_limit= DecodeRSSI(beacon.rfu1[1])
12      sf7_limit= DecodeRSSI(beacon.rfu1[2])
13      # Select SF
14      IF curren_rssi >= sf7_limit
15        new_SF= SF7
16      ELSE IF curren_rssi >= sf8_limit
17        new_SF= SF8
18      ELSE IF curren_rssi >= sf9_limit
19        new_SF= SF9
20      ELSE
21        # Original SF (e.g., SF10, SF11, SF12)
22        new_SF= previous_sf
23      END IF
24    END IF
25  END IF
26 END WHILE

```

Figure 5. End device pseudo-code for Gateway-based SmartSync LoRaWAN.

The main components of the proposed Gateway-based SmartSync system are depicted in Figure 6. It shows an abstraction of the primary procedures carried out within the gateway, starting with extracting the required information from each received uplink packet, such as the RSSI, the used SF, and the device ID. The gateway then tracks known devices using the recorder module and performs the proposed sensitivity balancing and beacon modification procedures. On the other hand, the general components within an end device illustrate the procedure of checking the Gateway-based SmartSync operation, decoding the received information, and adjusting its SF accordingly.

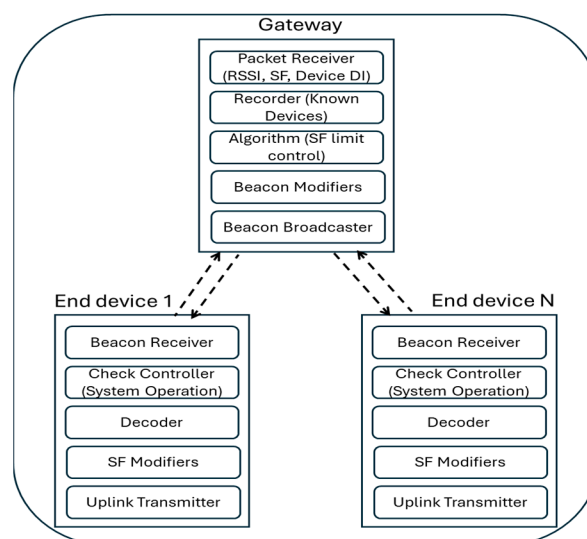


Figure 6. Gateway-based SmartSync system.

4. Evaluation

The adopted network topology is shown in Figure 7. The network consists of a network server, a LoRaWAN gateway, and LoRaWAN end devices. The gateway is linked

to the network server using a standard internet connection, whereas LoRa technology is used to communicate with other end devices. The simulated topology assumes a single LoRaWAN gateway covering the deployment area. This configuration allows the impact of the proposed mechanisms on SF distribution and uplink performance to be evaluated without the additional complexity introduced by multiple gateways. Devices are supposed to operate in class A and class B modes. The OMNeT++ [28] network simulator is used for the evaluation. The Inet framework [29] and Flora [30] project were used, which includes the implementation of LoRaWAN. The proposed systems were implemented within these frameworks to evaluate the proposals.

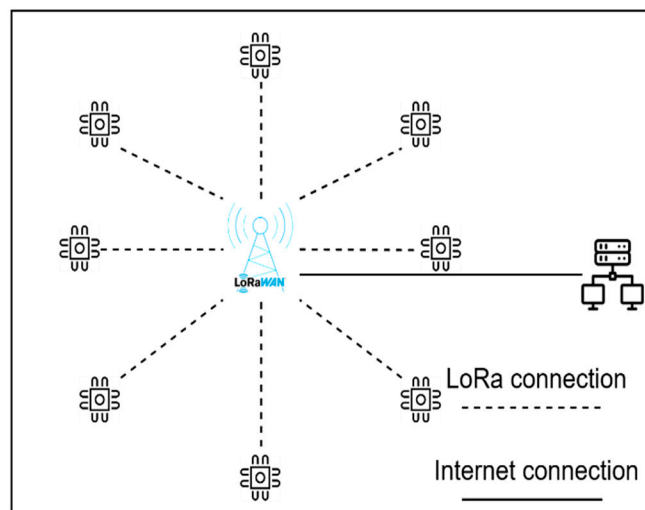


Figure 7. Adopted topology.

The evaluation process involves the use of a log-distance path loss model, where signal attenuation is calculated over distance based on the log-distance loss model. However, environmental obstruction, such as trees and buildings, is also considered, where a log-normal distributed variation is added to model real-world variability, as shown in Equation (7).

$$PL_d = PL_{d_0} + 10n \log\left(\frac{d}{d_0}\right) + x_g \quad (7)$$

PL_d is a path loss at distance d and PL_{d_0} is the reference value for path loss distance d_0 . The n parameter is the path loss exponent that depends on the environment and presents the effect of the distance on the data rate. x_g represents shadowing and is a zero-mean Gaussian random variable with g standard deviation. It is used to reflect flat fading attenuation, and it takes a 0 value if no fading exists.

The SFs are reconfigurable parameter ranges from SF7 to SF12. Each SF has a minimum RSSI value that must be met by the received signal in order to be demodulated successfully. Table 2 lists the SF sensitivity for the SX1272 LoRa device [31]. These values are used as the sensitivity thresholds for the ranges of the SFs within the proposed systems unless they are adjusted deliberately by the Gateway-based SmartSync system.

A simulation was conducted over a 25 km² area (5 km by 5 km) with varying numbers of end devices. A gateway equipped with the LoRaWAN class B protocol was positioned at the center of this area. All end devices were mobile, moving at a speed of 5 m/s, following the random waypoint mobility model with a random pause time ranging from 20 to 40 s. This model is widely used in network simulations due to its simplicity and ability to generate diverse movement patterns. It provides a generalized representation of mobile IoT devices with varying positions and link conditions. The selected speed of 5 m/s reflects moderate mobility scenarios, such as pedestrian movement and low-speed

vehicular applications, in urban environments. Each device sends an uplink message of 30 bytes once every hour. This low transmission rate is consistent with many LoRaWAN applications, such as environmental monitoring and smart metering, where devices operate under strict duty cycle constraints and transmit infrequently to conserve energy. The configuration parameters used in the simulation are presented in Table 3. At the start of the simulation, all devices operated using SF12, which was subsequently adjusted during the simulation based on the implemented system. The baseline approach applied the standard ADR mechanism, while the proposed systems utilized the newly introduced strategies. The threshold for triggering the Gateway-based SmartSync mechanism is set to 80.

Table 2. SF and receiver sensitivity with 125 kHz bandwidth.

Spreading Factor	Receiver Sensitivity
12	−137
11	−135
10	−133
9	−130
8	−127
7	−123

Table 3. Simulation parameters.

Name	Value
Network size	100–1000 in steps of 100
Area size	5 km by 5 km
Simulation period	86,400 s (1 day)
Bandwidth	125 kHz
Coding rate	4/5
Path loss model	LoRa log-normal shadowing
Frequency	868 MHz
Beacon period	128 s
Ping number	4
Transmission power	14 dBm
Packet size	30 bytes
Reptation	10

To ensure the reliability of the results, each simulation scenario was executed 10 times using different random seeds. The reported values represent the average across these runs, while the variability is illustrated through error bars in the figures, corresponding to the 95% confidence intervals. In addition, a two-sample t-test was conducted to evaluate the statistical significance of the performance differences between the proposed SmartSync systems and the baseline.

5. Results and Discussion

To evaluate the proposed systems, several key metrics were analyzed. These metrics assess the efficiency, network reliability, and scalability of the systems under diverse scenarios.

- **Packet Delivery Ratio (PDR):** this is a key metric used to identify the reliability of the system. It is defined as the ratio of total successfully received packets by the gateway to the total number of sent packets by all end devices.

$$PDR = \frac{\text{Number of Packets delivered}}{\text{Number of Packets sent}} \times 100 \quad (8)$$

- Collision: this metric is used to measure packet collisions that occur when more than one device transmits traffic simultaneously using similar channels and SFs. Such incidents lead to interference and packet loss. Note: this does not include the lost packet due to being received under sensitivity.
- Average energy consumption: the average consumption for a device is calculated as the total energy consumption for all devices divided by the number of devices participating in the networks. It includes all the device states, which are the sleep mode, processing mode, transmission mode, and reception mode. The energy consumption of end devices is evaluated using the built-in energy model of the FLoRa framework within OMNeT++. A device is equipped with a LoRaEnergyConsumer module, which follows a state-based energy model accounting for transmission, reception, processing, and sleep states. The energy consumed in each state is calculated based on the duration of that state and the corresponding power consumption.

$$\text{Average Energy Consumption} = \frac{\text{total consumption by all devices}}{\text{number of devices}} \quad (9)$$

- Below sensitivity: this metric is the total number of received packets by the gateway with RSSIs below the sensitivity threshold. The minimum receiver sensitivity for each SF is shown in Table 2. Such packets are considered lost and do not contribute to successful packet delivery. This type of loss is distinct from packet loss due to collisions or interference, as it is caused by insufficient signal strength rather than simultaneous transmissions.
- SF distribution: this metric describes the utilization of SFs over the total simulation period. It shows the proportion of successful transmissions over each SF.

Figure 8 shows a line chart of the successfully delivered packets for the three systems across different network sizes. The chart shows a decrease in the PDR for all three systems as the network size grows. This is expected as collision chances are higher due to increased collision probabilities in dense networks. Edge-based SmartSync and Gateway-based SmartSync maintain 80% and higher success ratios across all simulated network sizes, indicating the reliability of the proposed systems. In contrast, a noticeable drop in the efficiency of the baseline system is observed as the number of end devices increases in the network, where it falls to under 50% with a 1000 network size. This is likely due to traffic collisions and the lack of proper SF allocations. Gateway-based SmartSync achieves the best performance due to the implemented mechanism of balancing the number of devices on the lower SFs. The proposed systems improve the PDR by more than 40% compared to baseline with the largest network size, demonstrating the effectiveness of the proposed SF re-allocation mechanisms, as they prioritize lower SFs when signal conditions allow, especially in dynamic environments with mobile devices. The use of lower SFs means less time on air for transmitted packets and hence minimizes the chance of collisions. The improvement is statistically significant, as confirmed by a two-sample *t*-test, with *p*-values below 0.001, demonstrating that the performance gains of the proposed systems are highly robust across evaluated network sizes.

The number of collisions for baseline, Edge-based SmartSync, and Gateway-based SmartSync systems across the network sizes is presented in Figures 9 and 10. They highlight the total lost packets due to overlapping transmissions. Overall, collisions are observed to increase as the network size increases for all three systems, which reflects higher interference in dense networks. Gateway-based SmartSync and Edge-based SmartSync consistently exhibit fewer collisions compared to the baseline system, which experiences significantly more collisions, almost five times that of Gateway-based SmartSync, in a network with 1000 end devices. More collisions occurred in the baseline system, and a lower PDR

was achieved, as previously shown in Figure 8. Compared to the baseline system, both Edge-based SmartSync and Gateway-based SmartSync systems reduce collisions by about 80% for a dense network, which reflects the ability of the proposed solutions to improve network efficiency.

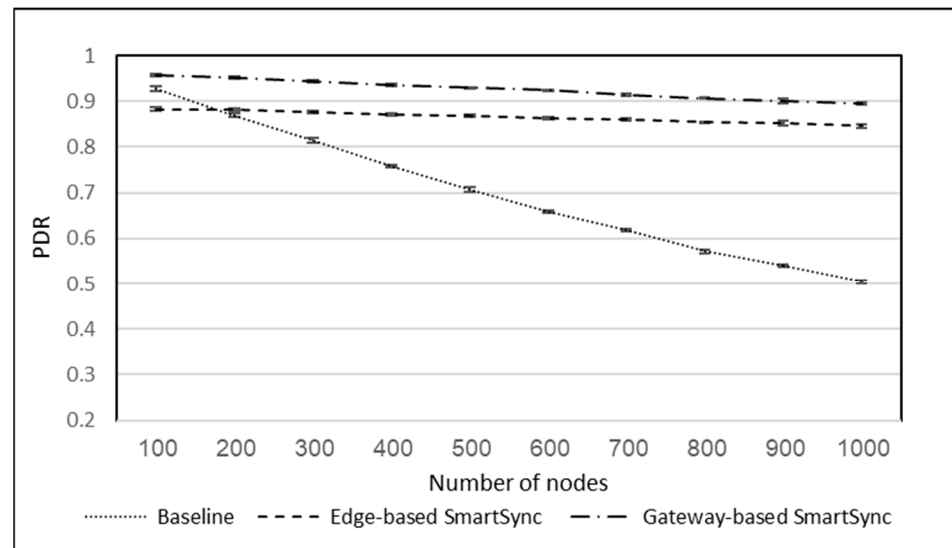


Figure 8. Packet delivery.

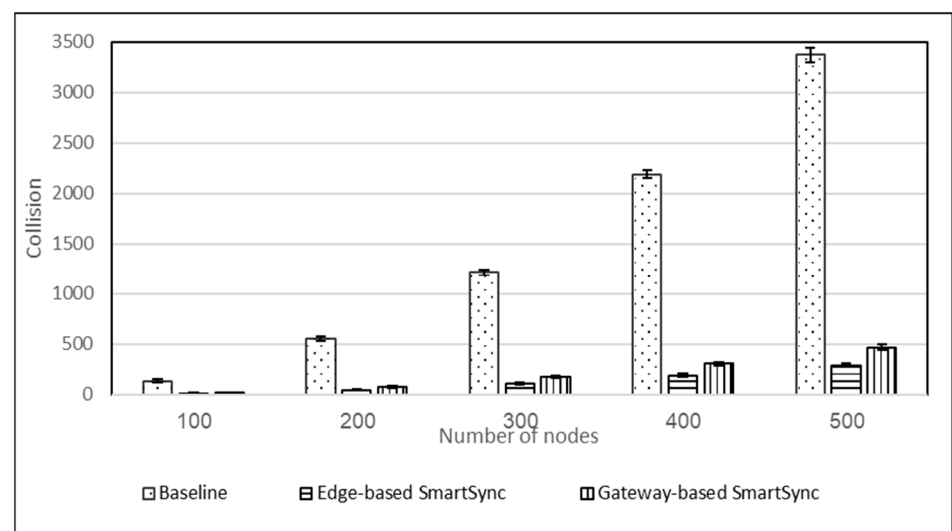


Figure 9. Collision 100–500.

Figure 11 shows a bar chart of average energy consumption per device for the three systems across all the simulated network sizes. The y-axis represents energy consumption in joules, including all the possible modes of devices: sleeping, processing, transmitting, and reception modes. Each of the three systems maintains a similar value of average energy consumption despite the changes in network size.

However, the baseline system exhibits significantly higher energy consumption, exceeding 8 joules, compared to approximately 5 joules for Edge-based SmartSync and Gateway-based SmartSync. The high energy consumption of the baseline system correlates with the high collision count and low PDR, plotted in previous figures. This is caused by inefficient SF allocation, as a higher SF involves higher time on air and longer radio transmission duration and hence consumes more energy. In contrast, the proposed systems efficiently allocate suitable SFs for an end device, which results in lower energy consump-

tion. Energy saving exceeds 35% compared to the baseline. Moreover, it is observed that the Edge-based system uses slightly less energy compared to the Gateway-based SmartSync system due to the light implementation mechanism.

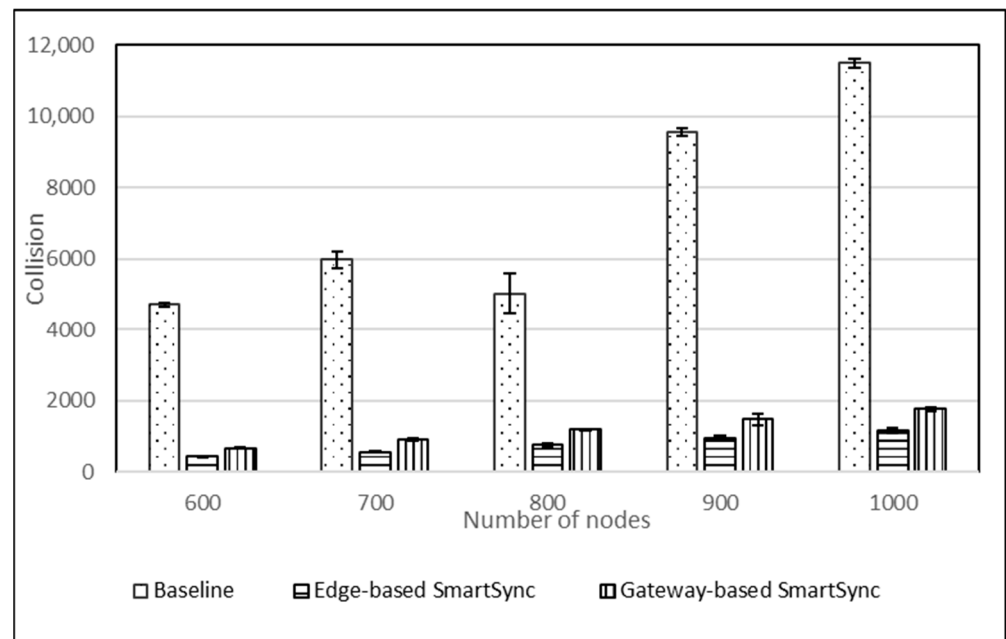


Figure 10. Collision 600–1000.

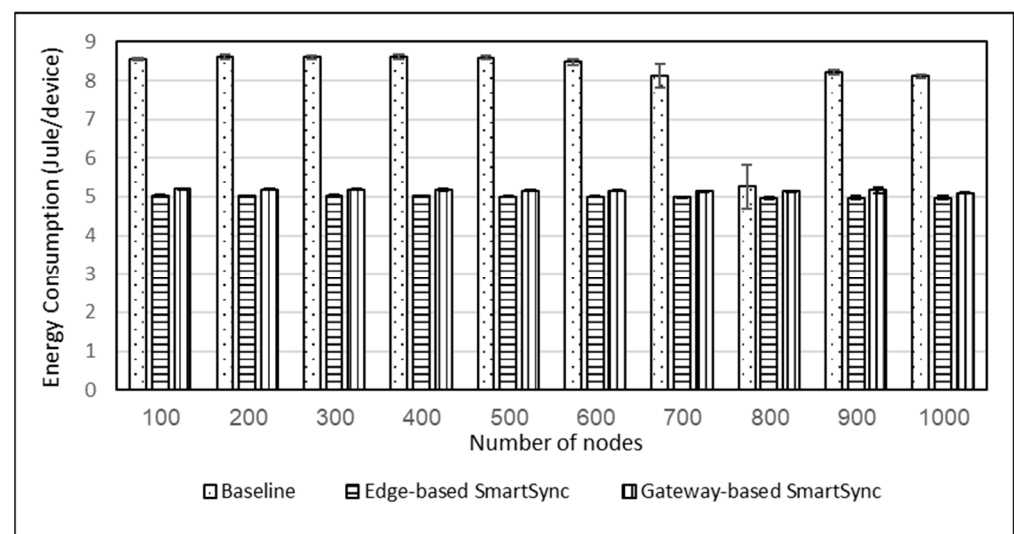


Figure 11. Energy consumption.

The total number of sensitivity-related packet losses across network sizes for the three systems is depicted in Figure 12. The baseline system exhibits the lowest losses despite its poor performance, as previously described in Figure 8, particularly with networks consisting of 500 devices or more. As Figures 9 and 10 show, the baseline system has the highest collision count, which primarily accounts for its poor performance. The lowest sensitivity-related packet losses are attributed to the use of a standard adaptive data rate technique that implies less frequent modification of SFs, with every 20 received messages.

As the figure illustrates, Edge-based SmartSync experiences more sensitivity-related losses compared to Gateway-based SmartSync. This is justified by the fact that devices change their locations frequently due to mobility and keep using previously allocated SFs. However, the proposed RSSI-based SF balancing in the Gateway-based SmartSync system

reduces sensitivity-related losses by approximately three times the sensitivity-related losses in the Edge-based SmartSync system.

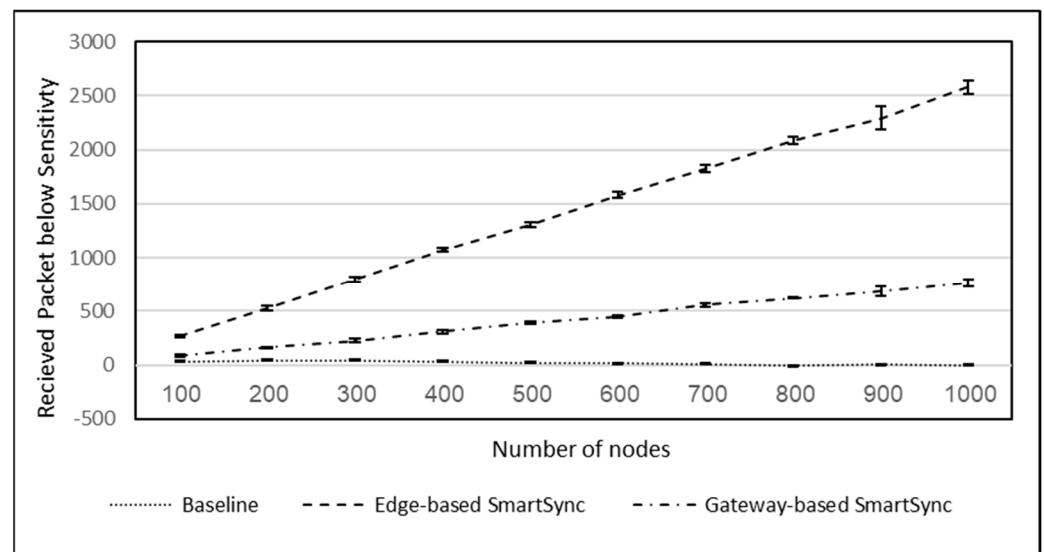


Figure 12. Received under sensitivity.

Figures 13–15 present pie charts of the SF distribution for the baseline, Edge-based SmartSync, and Gateway-based SmartSync systems in a network of 300 devices. The charts show the proportions of successful transmission for each SF. The result shows that the baseline system is heavily dependent on SF12, with approximately 95% of successful transmissions occurring with this SF. However, the heavy dependence on a high SF results in higher time on air, thereby increasing the likelihood of collisions. This was observed in previous results, where the baseline system achieved the lowest PDR and highest collision counts. In contrast, the Edge-based SmartSync system's mechanism for locally reallocating SFs based on the RSSI results in a dominant use of SF7. However, this dominance of SF7 achieves better results, contributing to a higher PDR (90%+), as shown in Figure 8. Finally, Figure 15 demonstrates that the Gateway-based SmartSync system effectively balances successful transmissions across SFs, particularly the lower ones, benefiting from its RSSI-based mechanism. This is also positively reflected in enhancing the PDR and reducing the average energy consumption of the proposed system.

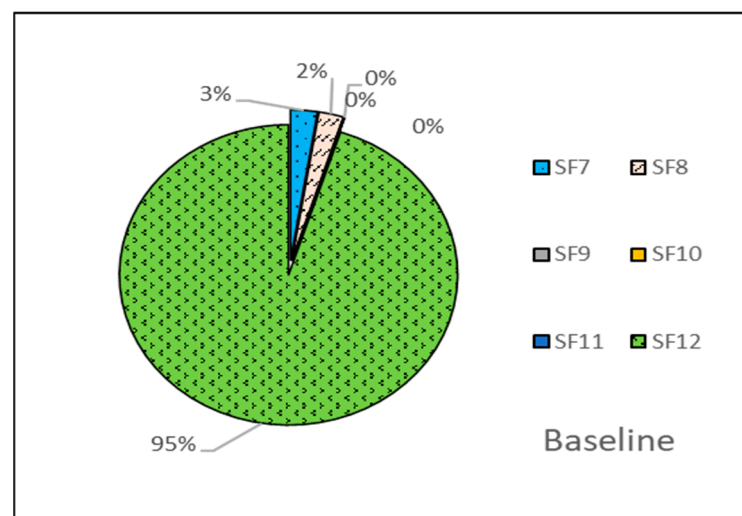


Figure 13. SF distribution for the baseline system.

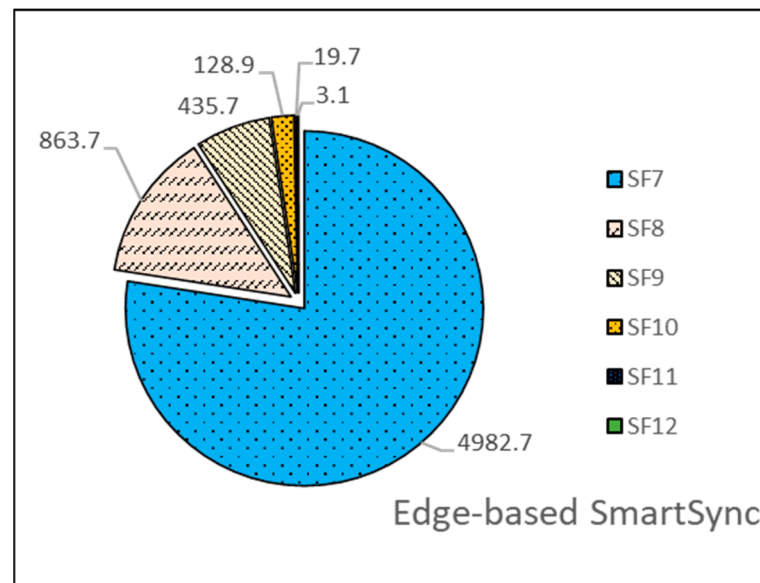


Figure 14. SF distribution for Edge-based SmartSync.

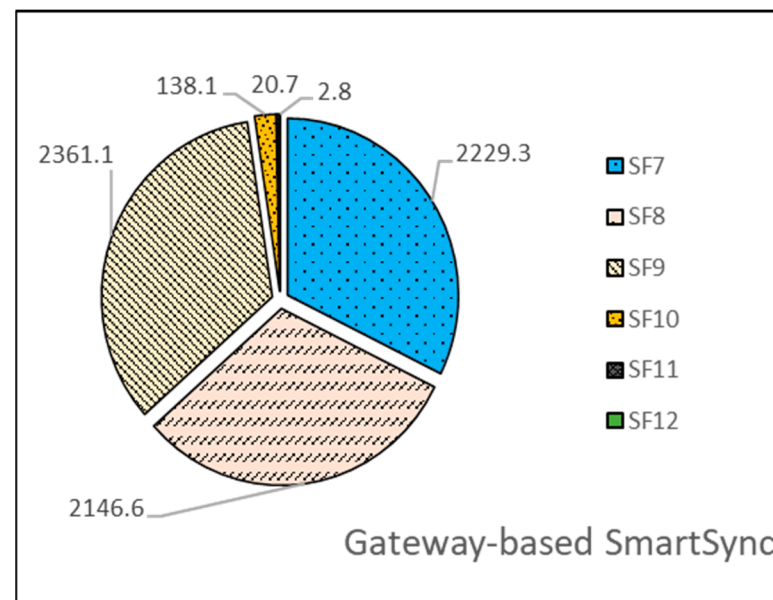


Figure 15. SF distribution for Gateway-based SmartSync.

6. Conclusions

This paper explores the potential of existing LoRaWAN class B mode features, proposing the Edge-based SmartSync and Gateway-based SmartSync systems to enhance the efficiency of LoRaWAN IoT applications. The Edge-based SmartSync system dynamically assigns the most suitable SF based on channel quality using freely broadcast beacons, potentially addressing the needs of mobile end devices. Channel quality information is retrieved by end devices from recently received beacons. The Gateway-based SmartSync system takes optimization further by reengineering beacons to convey device distribution across SF information, ensuring that end devices are fairly distributed across lower SFs to reduce collisions and enhance performance. This is achieved by gateway recording and tracking participating nodes using their uplink traffic, constructing sensitivity balancing limits, and distributing these sensitivity-related limits via beacon messages.

The systems were implemented in a network simulator, OMNeT++, and their performance was evaluated. The result demonstrates the ability of the proposed systems to

improve the efficiency of LoRaWAN. Both systems achieved an improved Packet Delivery Ratio and demonstrated scalability. In conclusion, the Edge-based and Gateway-based SmartSync systems represent a significant advancement in LoRaWAN technology, providing efficient and adaptable solutions for modern IoT challenges. The Gateway-based SmartSync mechanism for partitioning recorded RSSI values can be further investigated, with clustering algorithms that offer potential for defining natural boundaries for SF assignments. Future work will investigate the performance of the proposed systems in multi-gateway LoRaWAN deployments and evaluate the impact of gateway diversity on overall network scalability. In addition, the evaluation presented in this study is based on simulations. While simulations provide a controlled environment for investigating system performance, real-world deployments may introduce additional factors, such as environmental interference and gateway synchronization inaccuracy. Therefore, experimental validation through in-field deployment will be considered in future work.

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article is available at <https://doi.org/10.5281/zenodo.18870126>.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Jouhari, M.; Saeed, N.; Alouini, M.S.; Amhoud, E.M. A Survey on Scalable LoRaWAN for Massive IoT: Recent Advances, Potentials, and Challenges. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 1841–1876. [CrossRef]
2. Bonilla, V.; Campoverde, B.; Yoo, S.G. A Systematic Literature Review of LoRaWAN: Sensors and Applications. *Sensors* **2023**, *23*, 8440. [CrossRef] [PubMed]
3. Sigfox. Available online: <https://sigfox.com/> (accessed on 2 June 2025).
4. Narrowband—Internet of Things (NB-IoT). Available online: <https://www.gsma.com/solutions-and-impact/technologies/internet-of-things/narrow-band-internet-of-things-nb-iot/> (accessed on 3 June 2025).
5. Grant, S. 3GPP Low Power Wide Area Technologies. 2016. Available online: <https://www.gsma.com/solutions-and-impact/technologies/internet-of-things/wp-content/uploads/2016/10/3GPP-Low-Power-Wide-Area-Technologies-GSMA-White-Paper.pdf> (accessed on 3 June 2025).
6. Semtech Corporation. LoRa® and LoRaWAN®. 2024. Available online: <https://www.semtech.com/uploads/technology/LoRa/lora-and-lorawan.pdf> (accessed on 3 June 2025).
7. Maurya, P.; Hazra, A.; Kumari, P.; Sørensen, T.B.; Das, S.K. A Comprehensive Survey of Data-Driven Solutions for LoRaWAN: Challenges and Future Directions. *ACM Trans. Internet Things* **2025**, *6*, 1–36. [CrossRef]
8. Ballerini, M.; Polonelli, T.; Brunelli, D.; Magno, M.; Benini, L. NB-IoT Versus LoRaWAN: An Experimental Evaluation for Industrial Applications. *IEEE Trans. Ind. Inform.* **2020**, *16*, 7802–7811. [CrossRef]
9. ITU. ITU-T Recommendation Y.4480 Low Power Protocol for Wide Area Wireless Networks. 2021. Available online: <https://www.itu.int/rec/T-REC-Y.4480-202111-I/en> (accessed on 3 June 2025).
10. The Things Network. Device Classes. Available online: <https://www.thethingsnetwork.org/docs/lorawan/classes/> (accessed on 23 July 2025).
11. Todoli-Ferrandis, D.; Silvestre-Blanes, J.; Sempere-Paya, V.; Santonja-Climent, S. Adaptive Beacon Period Configurator for Scalable LoRaWAN Downlink Applications. *IEEE Access* **2023**, *11*, 83627–83638. [CrossRef]
12. Jagatha, A.; Rimal, B.P.; Vaidyan, V.; Wang, Y. Exploring LoRaWAN Class B and Class C Devices: Performance Analysis and Parameter Optimization Strategies. In *IEEE International Conference on Electro Information Technology*; IEEE: New York, NY, USA, 2024; pp. 194–201. [CrossRef]
13. Povalac, A.; Kral, J.; Arthaber, H.; Kolar, O.; Novak, M. Exploring LoRaWAN Traffic: In-Depth Analysis of IoT Network Communications. *Sensors* **2023**, *23*, 7333. [CrossRef]
14. Louati, R.; Thaalbi, M.; Tabbane, N. Enhancing LoRaWAN Class B: A Pseudo-Random Access Mechanism with Orthogonality Awareness. In *International Wireless Communications and Mobile Computing, IWCMC*; IEEE: New York, NY, USA, 2023; pp. 1454–1459. [CrossRef]
15. Todoli-ferrandis, D.; Silvestre-blanes, J.; Sempere-payá, V. Robust downlink mechanism for industrial internet of things using lorawan networks. *Electronics* **2021**, *10*, 2122. [CrossRef]

16. Ron, D.; Lee, C.J.; Lee, K.; Choi, H.H.; Lee, J.R. Performance Analysis and Optimization of Downlink Transmission in LoRaWAN Class B Mode. *IEEE Internet Things J.* **2020**, *7*, 7836–7847. [[CrossRef](#)]
17. Mroue, H.; Parrein, B.; Hamrioui, S.; Bakowski, P.; Nasser, A.; Cruz, E.M.; Vince, W. LoRa+: An extension of LoRaWAN protocol to reduce infrastructure costs by improving the Quality of Service. *Internet Things* **2020**, *9*, 100176. [[CrossRef](#)]
18. El Fehri, C.; Kassab, M.; Baccour, N.; Abdellatif, S.; Berthou, P.; Kammoun, I. An Uplink Synchronization scheme for LoRaWAN Class B. In *International Conference on Wireless and Mobile Computing, Networking and Communications*; IEEE: New York, NY, USA, 2019; pp. 47–52. [[CrossRef](#)]
19. El Fehri, C.; Baccour, N.; Kammoun, I. A New Schedule-Based Scheme for Uplink Communications in LoRaWAN. *IEEE Open J. Commun. Soc.* **2023**, *4*, 2815–2829. [[CrossRef](#)]
20. Al Mojamed, M. Beacon-Based Uplink Transmission for Lorawan Direct to Leo Satellite Internet of Things. *Int. J. Comput. Netw. Commun.* **2024**, *16*, 43–58. [[CrossRef](#)]
21. Fehri, C.E.; Kassab, M.; Baccour, N.; Abdellatif, S.; Berthou, P.; Kammoun, I. Evaluation of the use of class B LoraWAN for the coordination of distributed interface protection systems in smart grids. *J. Sens. Actuator Netw.* **2020**, *9*, 13. [[CrossRef](#)]
22. Chasserat, L.; Accettura, N.; Berthou, P. LoRaSync: Energy efficient synchronization for scalable LoRaWAN. *Trans. Emerg. Telecommun. Technol.* **2024**, *35*, e4940. [[CrossRef](#)]
23. Chasserat, L.; Accettura, N.; Berthou, P. Short: Achieving Energy Efficiency in Dense LoRaWANs through TDMA. In *IEEE 21st International Symposium on a World of Wireless, Mobile and Multimedia Networks*; IEEE: New York, NY, USA, 2020.
24. Elbsir, H.E.; Kassab, M.; Bhiri, S.; Bedoui, M.H. Evaluation of LoRaWAN Class B efficiency for downlink traffic. In *2020 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*; IEEE: New York, NY, USA, 2020; pp. 105–110. [[CrossRef](#)]
25. Finnegan, J.; Brown, S.; Farrell, R. Evaluating the Scalability of LoRaWAN Gateways for Class B Communication in ns-3. In *Institute of Electrical and Electronics Engineers*; IEEE: New York, NY, USA, 2018; p. 47.
26. Elbsir, H.E.; Kassab, M.; Bhiri, S.; Bedoui, M.H. Evaluation of LoRaWAN class B performances and its optimization for better support of actuators. *Comput. Commun.* **2023**, *198*, 128–139. [[CrossRef](#)]
27. Zorbas, D.; Fafoutis, X. Time-Slotted LoRa Networks: Design Considerations, Implementations, and Perspectives. *IEEE Internet Things Mag.* **2021**, *4*, 84–89. [[CrossRef](#)]
28. OMNeT++ Discrete Event Simulator. Available online: <https://omnetpp.org/> (accessed on 10 May 2025).
29. INET Framework for OMNeT++. Available online: <https://inet.omnetpp.org/> (accessed on 10 January 2024).
30. FloRa Framework. Available online: <https://flora.aalto.fi/> (accessed on 18 July 2025).
31. Semtech. Wireless & Sensing Products Datasheet. SX1272. Available online: https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/440000001NCE/v_VBhk1IolDgxwwnOpcS_vTFxPfSEPQbuneK3mWsXIU (accessed on 18 July 2025).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.