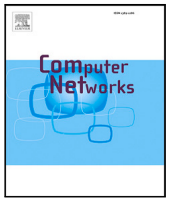
Computer Networks 221 (2023) 109525

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Efficient online resource allocation in large-scale LoRaWAN networks: A multi-agent approach

Celia Garrido-Hidalgo [a,*], Luis Roda-Sanchez [a,b], F. Javier Ramírez [c], Antonio Fernández-Caballero [a], Teresa Olivares [a]

[a] Albacete Research Institute of Informatics and Computer Systems Department, Universidad de Castilla-La Mancha, Albacete, 02071, Spain
[b] NEC Ibérica S.L., Madrid, 28108, Spain
[c] School of Industrial Engineering, Department of Business Administration, Universidad de Castilla-La Mancha, Albacete, 02071, Spain

ARTICLE INFO

Keywords:
LoRaWAN
Scheduling
Scalability
Resource allocation
Multi-agent system

ABSTRACT

The recent proliferation of the Industrial Internet of Things has revealed the potential of Low-Power Wide-Area Networks as a complementary solution to cellular technologies. In this context, the LoRaWAN standard has already been consolidated as one of the most extended technologies in academia and industry for lightweight machine-type communications under negligible energy and cost. As LoRaWAN's Aloha-like nature is known to hinder its reliability, especially under high-traffic and large-scale deployments, numerous time-slotted approaches have been presented as a means to schedule LoRa transmissions accordingly. However, the online allocation of resources based on application constraints has received scant attention in the literature, despite having proved to be significant in real-world deployments. To shed light on this question, this paper proposes a multi-agent approach to efficient resource allocation in multi-SF LoRaWAN networks, addressing architecture design, logic implementation and scalability-oriented evaluation. The integration of agents in the system resulted in network-size improvements of up to 21.6% and 66.7% (for nearby or scatter node distributions within the gateway, respectively). The work provides a set of learned lessons regarding *slot-length computation* and *end-node allocation* strategies enabling large-scale collision-free channel access in LoRaWAN networks.

1. Introduction

The continuous expansion of the Internet of Things (IoT) is expected to reach up to twenty-two billion connected devices by 2025, of which nearly ten percent will be governed by Low-Power Wide-Area Networks (LPWANs) [1]. While the aim of these is to complement cellular network deployments in a variety of domains [2], the role of LoRaWAN technology within the LPWAN ecosystem has received special attention from academia and industry, given its relatively low power consumption, deployment cost, and long range [3].

Resource allocation and customization have become key aspects in IoT for supporting specific quality of service (QoS) requirements, given the inherent network constraints, such as unreliable radio links or scarce resources to be shared, and a growing heterogeneity in terms of hardware diversity [4,5].

Despite the large body of literature addressing contributions to the LoRaWAN standard, there remain certain drawbacks related to scalability that need to be addressed given its Aloha-like nature and the need for low overhead. As a first approach to alleviating such reliability concerns, different lightweight strategies for the allocation of Spreading Factors (SFs) in dense LoRaWAN networks have been proposed in the literature, with a view to statistically minimizing the probabilities of two or more transmissions overlapping in time and frequency [6,7]. While their application does not typically increase the network's control overhead, reliability improvements will continue to be dependent on the overall size of the network being served. In contrast, scheduling mechanisms are aimed at increasing reliability based on the allocation of transmission slots at the expense of a certain increase in overhead, of which time-slotted approaches have generated particular interest towards collision-free channel access in dense LoRaWAN network deployments [8].

In this direction, the work presented by Haxhibeqiri et al. [9] introduced a low-overhead synchronization and scheduling algorithm to be built on top of LoRaWAN Class A end nodes which, more recently, was implemented and validated on top of STM32 hardware [10]. The results obtained served as a set of lessons learned in the context of real-world scheduling implementations, which pointed to the *coexistence of heterogeneous clock sources (clock diversity)* in the same network and the need for pre-loaded *static network configurations* at the network-server

* Corresponding author.
E-mail address: celia.garrido@uclm.es (C. Garrido-Hidalgo).

https://doi.org/10.1016/j.comnet.2022.109525
Received 8 August 2022; Received in revised form 23 November 2022; Accepted 11 December 2022
Available online 17 December 2022

List of Acronyms

ADR	Adaptive Data Rate
BP	Blackout Period
BW	Bandwidth
CR	Coding Rate
CSS	Chirp-Spread Spectrum
DC	Duty Cycle
DER	Deliver Extraction Rate
DR	Data Rate
ETSI	European Telecommunications and Standards Institute
gRPC	High-Performance Remote Procedure Call
IoT	Internet of Things
IPC	Inter-Process Communication
ITU	International Telecommunication Union
LoRaWAN	Long-Range Wide-Area Network
LPWAN	Low-Power Wide-Area Network
ML	Machine Learning
MAC	Medium Access Control
MAS	Multi-Agent System
MCU	Micro-controller Units
MQTT	Message Queuing Telemetry Transport
NSSE	Network Synchronization and Scheduling Entity
PDR	Packet Delivery Ratio
PER	Packet Error Ratio
QoS	Quality of Service
RSSI	Received Signal Strength Indicator
SE	Setup
SF	Spreading Factor
ST	Strategy
SNR	Signal-to-Noise Ratio
TDMA	Time-Division Multiple Access
VM	Virtual Machine

side as two key aspects to be further addressed so as to reach significant scalability improvements. The higher the required communications' robustness, the more conservative the criteria for establishing slot lengths and synchronization periods in the network should be, at the expense of reducing the channel utilization efficiency and, hence, the overall achievable network size under certain QoS conditions.

At this point, the design and integration of a multi-agent system (MAS) is proposed in this work to increase the flexibility of time-slotted LoRaWAN networks through an efficient allocation of resources that enables large-scale multi-SF deployments in compliance with specific QoS requirements. The proposed system extends previous algorithms by enabling end-device resource allocation based on application requirements (e.g. uplink periodicity or variable payload lengths), network resource availability (e.g. channel utilization or duty cycling), and physical constraints (e.g. clock diversity or distance to the gateway) in pursuit of collision-free channel access. This new network management component enhances the interaction of end nodes with Network Synchronization and Scheduling Entities (NSSEs) while being able to launch or terminate scheduling instances on demand that exploit the orthogonality of LoRa transmissions over different SFs at the gateway level. This work addresses the architecture design, system logic and implementation of the network, as well as its experimental validation through a set of emulated end nodes.

The paper is structured as follows: Section 2 presents the fundamentals of LoRaWAN and explores the related works from the literature, Section 3 provides a definition of the network metrics used, Section 4 presents the multi-agent approach, Section 5 describes the experimental system setup, results and discussion, and Section 6 highlights a set of conclusions and future works.

2. Background

This section first briefly describes the fundamentals of LoRa and LoRaWAN technologies in Section 2.1 as a means of providing a technical background on their principle of operation, strengths, and physical constraints. Second, Sections 2.2 and 2.3 review a set of relevant SF allocation and Time Division Multipla Access (TDMA) strategies from the literature designed to improve LoRaWAN scalability issues. Finally, the major contributions of this work to the existing literature are highlighted in Section 2.4.

2.1. LoRa and LoRaWAN fundamentals

The LoRaWAN specification [11] by the LoRa Alliance defines a Medium Access Control (MAC) of LoRa's physical (PHY) layer transmissions patented by Semtech [12], which are based on chirp-spread spectrum (CSS). Data rates (DR) achieved by LoRa signals are, therefore, conditioned by the bandwidth (BW), coding rate (CR), and SF used. The lower the SF, the higher is the data rate at the expense of a more limited communication range.

LoRa packets comprise a preamble for synchronization between the receiver and transmitter, an optional header, and the frame payload itself. In order to effectively comply with regulatory constraints, the time on air of scheduled LoRa transmissions needs to be computed, based on the symbol duration (T_{symb}), that is, the time required to send 2^{SF} chirps. The duration of a packet's preamble ($T_{preamble}$) and payload ($T_{payload}$) will result in the time on air of a LoRa packet (T_{tx}), as defined in Eqs. (1) and (2), following to Semtech's modem design guide [12]:

$$T_{preamble} = (n_{preamble} + 4.25) \times T_{sym} \tag{1}$$

$$T_{payload} = P_{symb_{Nb}} \times T_{symb} \tag{2}$$

where $n_{preamble}$ is the number of preamble symbols, T_{symb} is the symbol duration ($2^{SF}/BW$) and $P_{symb_{Nb}}$ is the number of symbols that make up the packet's payload and header [12].

LoRa signals are known to be quasi-orthogonal, which means that they are not expected to collide when transmitted over the same channel and BW using different SFs. In this regard, several studies have explored to which extent orthogonal LoRa transmissions can be successfully demodulated. Nevertheless, in terms of network capacity, no noticeable differences have been found [13].

LoRaWAN is based on a star-of-stars topology, where gateways are responsible for forwarding LoRa packets from end devices to a specified network server via a back-haul such as Ethernet or 4G/5G. In Europe, LoRaWAN operates in the 433, 868 and 915-MHz unlicensed frequency bands, of which the 868-MHz is currently regulated by ETSI's EN300.220 standard. Hence, duty cycles are restricted to a maximum of 1% in all sub-bands, except for g2 (868.7–869.2 MHz) and g3 (869.4–869.65 MHz), where the maximum duty cycles are 0.1% and 10%, respectively. Despite gateways typically operating in the g3 sub-band, downlink communication is not encouraged in order to sustain the network's capacity, especially in single-gateway deployments. Furthermore, the half-duplex nature of gateways means that no uplink packets will be received during downlink transmission, which again significantly impacts scalability.

The time window for which a LoRaWAN device is unable to access a channel due to duty cycle restrictions after the transmission is typically known as *blackout period* (BP [14]), which is specified for gateways in Eq. (3).

$$BP_{DC=10\%} = \frac{T_{tx}}{DC} - T_{tx} = 9 \times T_{tx} \tag{3}$$

where DC is the downlink duty cycle, and T_{tx} is the transmission time.

The LoRaWAN standard defines Classes A, B, and C as a means to provide a trade-off between battery requirements and constraints imposed by the application being pursued. Of these, LoRaWAN Class A is mandatory to be supported by any LoRaWAN-compliant end device. It achieves the lowest power consumption by keeping end devices in sleep mode, which will only open two short receive windows upon uplink completion (after one and two seconds, respectively). Consequently, downlink traffic should be accordingly scheduled at the network-server side.

2.2. SF allocation in LoRaWAN networks

End devices located close to a gateway could transmit using any SF due to the small link budget required, although the lowest SF is encouraged so as to minimize power consumption and time-on-air usage. To do this, LoRaWAN implements an adaptive data rate (ADR) mechanism, which improves the network capacity by assigning the lowest-possible SF to end devices according to their received signal-strength indicator (RSSI) and signal-to-noise ratio (SNR) [11]. While this mechanism minimizes channel utilization and, hence, the energy required by all end nodes in the network [15], there exist different related drawbacks in terms of reliability. For instance, large-scale networks in which all end devices are located close to the central gateway will suffer from high intra-SF interference, which could be alleviated with a fair SF allocation strategy leveraging orthogonality across different SFs.

In this context, various alternatives to LoRaWAN's legacy ADR have been proposed in the literature which, instead of focusing on individual link-level performance to provide a suitable SF, consider the entire set of nodes in the network for fair SF and power allocation [16,17]. Adelfadeel et al. [18] proposed one of the first contributions in this direction, which balanced collision probabilities in the network regardless of the distance or link quality associated to end nodes to achieve fairness. Similarly, one of the most relevant contributions to SF allocation in LoRaWAN networks was provided by Cuomo et al. [6], who introduced *EXPLoRa* and *EXPLoRa-AT* strategies. The former consists of balancing the total number of end devices across available SFs while, the latter, provides a more sophisticated approach by equalizing the available time-on-air per SF. Both alternatives outperformed LoRaWAN's legacy ADR during high network loads through the addition of simple heuristics. Two additional heuristic-based approaches to SF allocation are *equidistant SF allocation* and *equal-area allocation*, which were validated by Duda and Heusse [7] in terms of Packet Delivery Ratios (PDR).

Two agent-based approaches to LoRaWAN-based SF allocation exist to date in the literature. In [19], the authors propose an intra-slicing resource-allocation technique as an alternative to legacy SF allocation implemented by ADR, which is evaluated using 4 end devices and a LoRaWAN gateway. Similarly, but in a simulator environment consisting of 30 nodes, the authors in [20] proposed a deep reinforcement learning system leveraging agent-based interaction for improving the legacy ADR schema through SF allocation.

With a view to addressing the so-called *capture effect* in LoRaWAN networks, Caillouet et al. [13] presented an algorithm that considered potential channel interferers for resource allocation to achieve improved scalability under fair reliability. Hamnache et al. [21] provided an approach to load-balancing across different SFs so as to sustain the networks' deliver extraction rate (DER). Following a similar approach, Sallum et al. [22] proposed a method to improve the QoS of LoRaWAN networks by assigning, on-demand, the optimal SF and carrier frequency to minimize collisions, which was resolved through mixed integer linear programming. By using the LoRaSim simulator, DER improvements of up to 6.6% were achieved. The same simulator was used in [23] to evaluate the reliability of APRA, a priority-aware bandwidth selection mechanism, which achieved packet error ratio (PER) reductions of up to 50% with respect to EXPLoRa-AT.

While the previous works succeeded in different ways in achieving capacity improvements in LoRaWAN networks through fair SF allocation, most of them were developed on the basis that information about the number, location and condition of nodes in the network is known in advance. Furthermore, downlink scalability received scant attention, which is known to limit to a great extent the achieved cell's capacity upon the re-allocation of physical-layer parameters.

2.3. Time-slotted LoRaWAN communications

LoRaWAN's band-usage constraints, especially in downlink, increase the difficulty of network management tasks, which should additionally guarantee low control overhead from the network to the field devices [24]. LoRaWAN Class B and C approaches have demonstrated significant improvements in terms of low-overhead device synchronization with respect to SF allocation strategies (which are aimed at minimizing but not avoiding collisions in the network) [25,26], although greater efforts are being made with Class A – mandatorily supported by any device – to reach an effective scheduling solution that does not comprise downlink scalability.

In this sense, TDMA provides a satisfactory approach to increasing the throughput in dense network deployments under low energy consumption, which has been validated in both simulation and experimental studies. Chasserat et al. [27] demonstrated a maximum achievable throughput lower than 18% in dense LoRaWAN networks (deployments consisting of 800 pure Class A end devices per unit of area), which increased up to 34% when implementing a TDMA schema (even for network densities of up to 1600 devices per unit of area). Moreover, significant improvements in terms of energy efficiency were identified as well, given the minimization of frame retransmissions due to collision avoidance.

With regard to TDMA-inspired approaches focused on Class A specification, let us highlight three relevant simulation studies from the literature: (i) Abdelfadeel et al. [28] demonstrated the benefits of exploiting SF orthogonality by using the LoRaFREE simulator; (ii) Reynders et al. [29] proposed a two-step scheduling mechanism in which nodes selected their preferred SF, channel and transmission power based on gateway parameter provisioning; and (iii) the authors in [30] addressed the NS-3 design and implementation of temporal maps in which nodes were assigned to orthogonal schedules. In this study, for the sake of simplicity, slot lengths were considered fixed and multiples of each other. The three studies support the benefits of time-slotted scheduling of transmissions in Class A LoRaWAN networks in terms of reliability and energy efficiency. Nevertheless, TDMA-based scheduling is only encouraged for dense network deployments, since a trade-off exists between the energy overhead required for the synchronization of LoRaWAN beacons and the achievable energy savings due to such collision avoidance.

As part of the findings from TDMA-oriented LoRaWAN, (i) *downlink synchronization overhead* and (ii) *clock diversity* were highlighted as the two key bottlenecks limiting the actual network capacity in [10,31], and [32].

In the first domain, different approaches to low overhead exist. First, the authors in [33] proposed replacing device synchronization with collision prediction which, using NS-3, resulted in twice the maximum network size for the same reference PDR. However, in this case, the overhead increased significantly after reaching a certain network size because of the re-allocation of nodes. Second, Zorbas et al. [34] proposed an offline time-slotted scheduling approach that minimized data collection periods to comply with downlink duty cycle constraints. Finally, Haxhibeqiri et al. [9] focused on low-overhead scheduling and synchronization, which was first validated in terms of reliability and energy consumption using the NS-3 simulator (the battery capacity used to perform synchronization was of 3 mAh, much lower than the required for re-transmissions of collided packets in unsynchronized

setups) and, more recently, implemented on top of real hardware with measured PDR improvements of up to 29% with respect to Aloha [10].

The co-existence of heterogeneous clocks has received scant attention in the literature, since not only the clock skew of nodes, but also their transmission or synchronization periods, are typically assumed constant. While this lack of flexibility can comprise the achievable network size under certain QoS requirements, enabling adaptive schedules might result in an increased network management complexity hindering in turn downlink performance.

2.4. Paper contributions

This work represents an extension of the low-overhead scheduling mechanism presented in [9] and validated in [10] to enhance the join mechanism of nodes through the design, development and testing of multi-agent network management for efficient online resource allocation in multi-SF deployments, which computes scheduling metrics online based on the network knowledge, hardware and physical constraints. The main contributions of this work are:

(i) The design and practical implementation of a multi-agent-based network management architecture for flexible allocation of resources in multi-SF LoRaWAN networks, the system logic design of which enables: (i) low-overhead joining of end devices, (ii) collision-free scheduling of transmissions, and (iii) goal-oriented allocation of devices within orthogonal schedules for improved scalability. The resulting end-to-end system serves as a modular benchmark and abstraction layer for the experimental scalability-oriented evaluation of novel SF allocation strategies on top of time-slotted LoRaWAN communications guaranteeing large-scale robustness.

(ii) End-to-end system implementation and validation through the proposal of various decision-making criteria dependent on live network metrics, where the performance of different *slot-length computation* and *end-device allocation* strategies is assessed in terms of network size improvements. In contrast to previous works from the literature where clock skews of joining devices are considered constant and known in advance, the integration of multi-agent network components is leveraged in this work to deal with the co-existence of heterogeneous clock sources in the network and application constraints such as variable payload lengths or radio-links qualities. To the best of our knowledge, this work presents the first multi-agent approach to LoRaWAN scheduling of transmissions demonstrating significant capacity improvements through efficient goal-oriented resource allocation.

(iii) Outline of a set of learned lessons for large-scale and application-oriented LoRaWAN deployments implementing time-slotted communications to support collision-free channel access in large-scale networks. While the TDMA-based collision avoidance mechanism implemented is aligned with previous works from the literature, the results obtained upon experimental validation of scalability-oriented criteria in constrained environments serve as a reference for hyper-parameter tuning in the training of network congestion prediction models.

Table 1 provides a comparative of the contributions of this work with related works from the literature in the domain of LoRaWAN-oriented resource allocation for improved capacity in large-scale network deployments.

3. Network metrics

This section describes the reference parameters used in this work to define slot lengths under a TDMA approach, while channel occupancies and downlink usages are selected as input to the decision-making criteria to be applied by the resource-allocation module. Despite all definitions being provided for the target scheduling algorithm, they can also be generalized for alternative time-slotted LoRaWAN schemes.

3.1. Slot length definition

The proposed slot-allocation method leverages SF orthogonality, since symbols transmitted simultaneously over the same channel can be received without interference provided they use different SFs [13]. Accordingly, it is necessary to define a specific slot length for each of the SFs and channels being used. Specifically, a minimum of six time-slotted schedules (one per SF) are assumed to be running simultaneously at gateway level. Slot lengths can be computed according to Eq. (4):

$$S_{len} = T_g + T_{tx} + \epsilon \tag{4}$$

where S_{len} is the slot length, T_g is the guard time required to compensate clock desynchronization over time (clock drift or skew), T_{tx} is the transmission time required at a specific SF, and ϵ is the existing time offset between each clock and the global time reference (all of them in the order of milliseconds). In this work, ϵ is assumed as negligible, since a previous experimental evaluation demonstrated no significant impact on communication reliability for different time offsets [10].

The first term required to compute slot lengths in Eq. (4) is the guard time, defined as the minimal time interval that needs to be added to a specific time slot to avoid transmission overlaps due to node desynchronization over time. It is, in turn, a function of the difference between the fastest and slowest nodes' clock skew in parts per million (ppms) and the defined synchronization period. Considering both positive and negative time drifts, that is, faster and slower clock deviations, the guard time is defined according to Eq. (5):

$$T_g = \left\lceil \frac{2 \times T_{sk} \times T_{syn}}{10^6} \right\rceil \tag{5}$$

where T_g is the guard time, T_{sk} is the clock skew, and T_{syn} is the synchronization period, which is defined as the time interval between two synchronization requests that ensures no overlapping between neighbour transmissions (based on the established guard times per slot).

The second term in Eq. (4), the transmission time (T_{tx}), is the time on air required for a LoRa transmission (defined according to Semtech's modem design guide [12] in Eq. (6)). It is a function of the transmitted payload length (PL) and the SF used. As may be noticed, the clock skew of each device is hardware-dependent, while payload lengths are application-specific. These, hence, represent fixed parameters to be considered during the definition of the slot lengths.

$$T_{tx} = T_{preamble} + T_{payload} \tag{6}$$

The transmitting SF and the synchronization period are configurable to a certain extent according to varying network conditions. Therefore, these are used to dynamically re-configure slot lengths to adapt each joining node to the existing network schedules with no need to re-launch new scheduling schemes. This represents a significant improvement with respect to previous works where synchronization periods are fixed and assumed to be known by all the components in the network. It is worth noting, however, that a minimum threshold ensuring low control overhead should exist, which is further detailed in Section 5 as part of the design criteria.

3.2. Network metrics for decision-making

While the previous metrics are required by the end nodes and the scheduler to initiate time-slotted communication, below we describe the formulae used to compute channel occupancies and downlink usages.

Adapting the TDMA spectrum resource definitions provided by the International Telecommunication Union (ITU) [36] to the technology-specific requirements of LoRaWAN (SF orthogonality and downlink channel-sharing), the occupancy time (T_0) is defined as the percentage of time that needs to be reserved in all the concurrent schedules (one

Table 1
Contextualization of this work and main contributions to the literature.

Source	Method	Target approach			Constraints considered			
		SF allocation	Time-slotted	Online[a]	Class A	Clock diversity[b]	Variable payload	Downlink[c]
[15]	Simulation	✓		✓	✓			
[17]	Simulation	✓		✓	✓			
[18]	Simulation	✓			✓			
[6]	Simulation	✓			✓			
[19]	Experimental	✓			✓			
[20]	Simulation	✓			✓			
[13]	Theory	✓			✓			
[21]	Simul./Experim.	✓			✓			
[22]	Simulation	✓		✓	✓			
[23]	Simulation	✓		✓	✓			
[25]	Simulation	✓	✓	✓				
[26]	Simulation		✓	✓				
[35]	Simulation	✓	✓	✓			✓	
[27]	Simulation		✓	✓				
[28]	Simulation	✓	✓	✓	✓			
[29]	Simulation	✓	✓	✓				✓
[30]	Simulation	✓	✓	✓				
[31]	Experimental		✓	✓	✓			✓
[32]	Simul./Experim.		✓	✓		✓		✓
[33]	Simulation		✓	✓	✓			✓
[34]	Simulation	✓	✓				✓	✓
[9]	Simulation		✓		✓			✓
[10]	Experimental		✓		✓			✓
This work	Emulation[d]	✓	✓	✓	✓	✓	✓	✓

[a]The nodes' deployment conditions are unknown to the resource-allocation system.
[b]The coexistence of heterogeneous clock sources is considered in evaluation.
[c]Downlink signalling required for resource-allocation is considered in evaluation.
[d]End nodes were emulated (real-world validation available in [10]), the MAS was validated experimentally.

per SF) for a specific node's uplink and downlink communication. It includes, therefore, the number of slots assigned for periodic uplink reports at a specific SF ($T_{0_{UL}}$), and the sum of time slots reserved at independent SFs for downlink communication ($T_{0_{DL}}$). It is computed according to Eq. (7):

$$T_0 = T_{0_{UL}} + T_{0_{DL}} = \left(\frac{S_{len}}{T_{UL}} + \sum_{i=SF7}^{SF12} \frac{\left\lceil \frac{S^*_{len}}{S_{len_i}} \right\rceil S_{len_i}}{T_{syn}} \right) \times 100 \tag{7}$$

where $T_{0_{UL}}$ is the time interval during which a specific channel is reserved by the device for uplink communication, $T_{0_{DL}}$ for downlink communication, T_{syn} is the synchronization period at a specific SF, T_{UL} is the uplink period required by the end device, S^*_{len} is the slot length at a specific SF-based schedule (equal to S_{len} or $\frac{S_{len}}{2}$ for best-case and worst-case scenarios, respectively), and S_{len_i} is the slot length at SF equal to i.

The second term, U_{DL}, is the downlink usage, that is, the time percentage that downlink will be unavailable due to an ongoing transmission of a specific node or the associated black-out period. It is, thus, computed as the sum of downlink usages of each end device n at a specific gateway listening to simultaneous SF schemes (U_{DL}), according to Eq. (8).

$$U_{DL} = \sum_{n=0}^{N} \frac{T_{tx_n} + BP_n}{T_{syn_n}} \times 100 \tag{8}$$

where T_{tx_n} is the time on air of node n required for transmission at the scheduled SF, BP_n is the *blackout period* of node n using such SF, and T_{syn_n} is its synchronization period.

Fig. 1 shows an example of three simultaneous gateway-level schedules in which the impact of slot reservation on channel occupation can be observed. Considering SF7 schedule from Fig. 1 ($NSSE_1$), when the end device ED_1 is expected to trigger a re-synchronization request ($synch_{req}$), slots need to be reserved simultaneously at the remaining schedules ($NSSE_2$ and $NSSE_3$) so that the downlink frame is successfully transmitted from the gateway to the concerned end device ($synch_{rep}$). The amount of time that needs to be reserved at the other SFs is shaded in grey, which is included in $T_{0_{DL}}$ according to Eq. (7). Note that guard times have not been represented in Fig. 1 for better legibility.

4. Multi-agent-based network resource allocation

In order to contextualize the multi-agent network components and resource-allocation method presented in this work (Sections 4.1 to 4.4), the logic behind the scheduling algorithm presented in [9] is first briefly described. It is built on top of LoRaWAN Class A stack, with end devices being expected to trigger communication at a specific SF by sending a synchronization request to the NSSE. Once the request is processed, the NSSE will provide a set of available slots where the concerned node is able to transmit data. It is encapsulated using probabilistic data structures –Bloom filters– in order to keep payload lengths to a minimum. The requests are eventually processed by the end device to be synchronized in time with the current schedule.

4.1. System architecture and logic

Fig. 2 depicts the system architecture, which is based on three reference layers : (i) the *perception layer*, composed of a set of constrained devices intended to send application-specific sensor data reports; (ii) the *network layer*, responsible for de-duplicating and forwarding LoRa traffic via a back-haul, such as Ethernet, 4G or 5G; and (iii) the *application and service layer*, which includes the components required for efficient resource management, synchronization, and scheduling of communications. This layer is conceived as a LoRaWAN network management module responsible for improved communications reliability and efficient use of resources, not limiting the integration of other technology-specific modules as proposed by Moons et al. [37].

The network layer follows the standard architecture of ChirpStack's LoRaWAN network server [38], where one or more distributed LoRaWAN gateways forward uplink data frames received from end devices to the remaining network-layer components –*gateway bridge*, *network server* and *application server*– by running Semtech's UDP packet

Fig. 1. Three schedules running in parallel at gateway level.

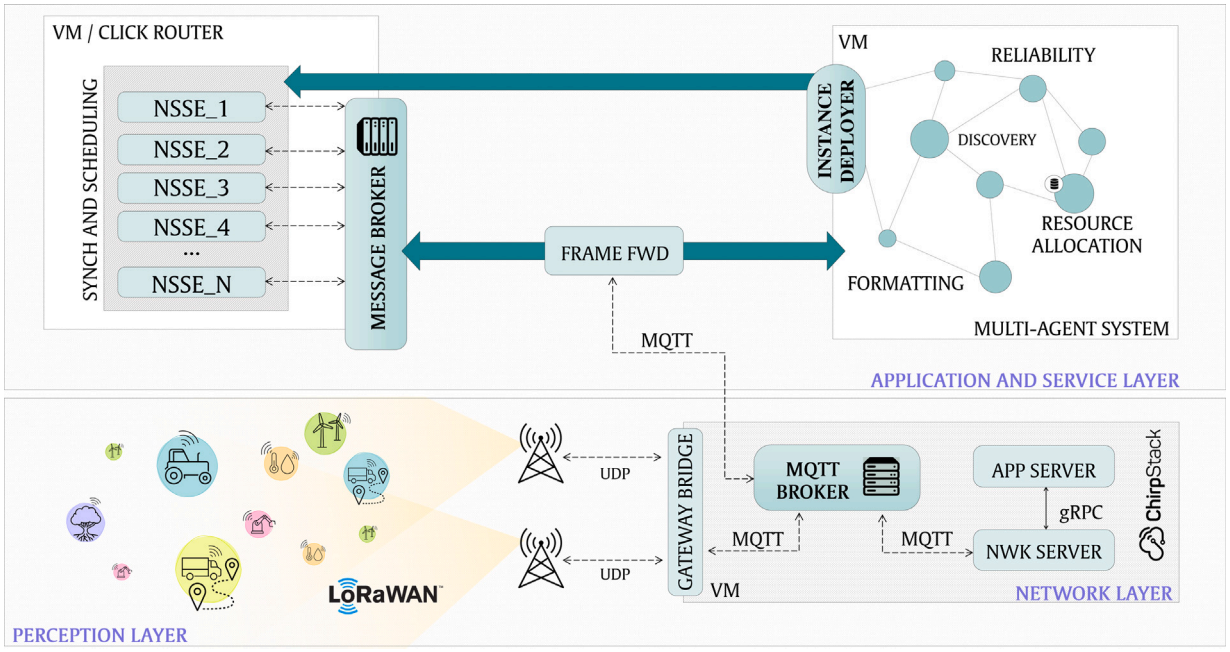


Fig. 2. End-to-end architecture for multi-agent-assisted synchronization and scheduling of LoRaWAN transmissions.

forwarder. The gateway bridge is responsible for translating the packet forwarder protocol format to ChirpStack's data formats, which connects to the network server via Message Queuing Telemetry Transport (MQTT). The latter, in turn, processes legacy LoRaWAN join requests, de-duplicates traffic received from multiple gateways, and manages downlink queues according to gateway availability. Finally, the network server establishes gRPC (high-performance Remote Procedure Call) inter-process communication (IPC) with the application server, which performs data-payload decryption/encryption tasks and provides integrations for application-specific purposes. At this point, the MQTT integration is used for traffic forwarding to the application and service-layer components.

Between network-layer and service-layer components sits a frame forwarder, which is responsible for filtering uplink packets and forwarding them to two ends: (i) the *network resource allocation* components, or (ii) the *NSSE*. While the latter is responsible for performing low-overhead synchronization and scheduling tasks on top of the LoRaWAN MAC layer (validated in practice in our previous work [10]), the former provides a modular engine for resource-aware decision-making upon device discovery. The service and application layer includes a database for context registration and application-specific payload storage and management.

NSSE instances are launched and terminated on demand for each of the SFs based on end-device requirements and available network resources at the gateway level. Conversely, the network resource allocation components – which in turn include an MQTT message broker for external communication – centralize resource allocation tasks and act as an abstraction layer for hardware-agnostic device registration.

The procedure to adapt each joining device to the most appropriate schedule at run time is shown in Fig. 3, and consists of the following steps:

(i) End devices initiate signalling with the backend-side to manifest their intention to join the network (not to be confused with LoRaWAN legacy's join-request) by providing their *thing ID* (group of devices sharing common application-specific constraints), *uplink period* (data-reporting period to be used), *clock skew* (hardware-specific frequency deviation in ppms), and *roaming indicator* (0 for static nodes, and 1 for mobile nodes).

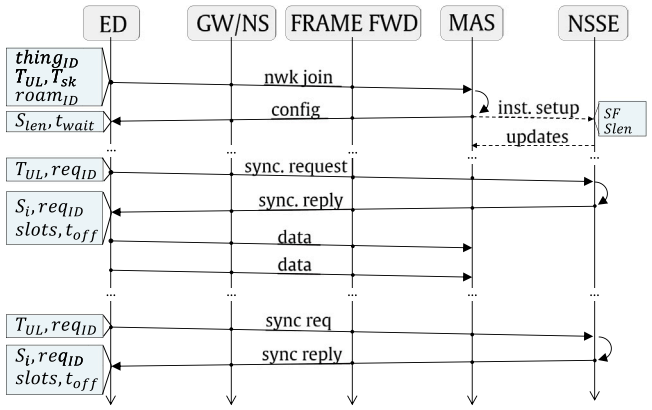


Fig. 3. End-device joining mechanism for resource-allocation in TDMA.

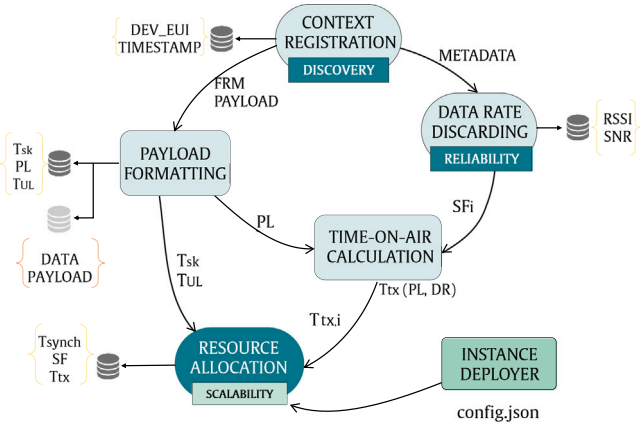


Fig. 4. Communication flows of the multi-agent network components.

(ii) The frame forwarder (see Fig. 2) will queue device-specific requests to the multi-agent resource allocator based on the received frame port.

(iii) Agents will first collect device-specific metadata to perform context registration and, then, negotiate in order to provide joining devices with the most appropriate configuration according to application requirements, hardware constraints, current network status, and availability of resources (uplink and downlink channel occupancies). This configuration consists of a preferred time window for triggering synchronization with the assigned NSSE, a transmitting SF and a synchronization period.

(iv) Having received the desired configuration, end nodes will provide their already-assigned NSSE with their *uplink period* and *request ID*. As the clock skew is not expected to change for each end device, it will be automatically linked to a device extended unique identifier (EUI) after context registration (previous step). Note that, provided application-specific constraints – location or payload size – do not change, future re-synchronizations will be automatically directed to the same NSSE instance.

(v) After processing this information, the NSSE will queue a response including the assigned scheduling parameters: *assigned slots* (through Bloom-filter probabilistic data structures), *request ID*, *slot index*, and *time offset* (at current slot index). This information will finally be processed by the device which will be synchronized in time and schedule with its respective NSSE.

4.2. Multi-agent network management module

The proposed network management module provides an intermediate negotiation with joining devices and NSSE (see Fig. 2), the aim of which is to improve the network capacity in TDMA-based LoRaWAN. The logic behind this system is presented in this section, as well as a brief definition of the agents' goals and tasks to be performed.

Fig. 4 depicts the system's communication flows. Despite each agent following a set of individual objectives, they all have a common goal, namely, to compute network metrics at the gateway level and, based on these, to provide each joining device with a synchronization period, transmitting SF, and slot length (see step (iii) in Section 4.1). As a result, an efficient access control scheme to increase the maximum number of devices being served by a single gateway is achieved.

The system's architecture is service-based, which enables an efficient update, integration and removal of modular components. It is composed of (but not limited to) six autonomous agents that interact with one another being, in turn, easily deployable as microservices. The following software agents are included: (i) *Device Registration* is responsible for device discovery; (ii) *Payload Formatting* for frame payload decoding and format adaption; (iii) *Data Rate Discarding* for the adoption and implementation of reliability criteria; (iv) *Time-on-Air Calculation* for packet air time computation; (v) *Resource Allocation* for device-oriented decision-making; and (vi) *Instance Deployer* for launching and termination of NSSE instances. These agents collaborate to perform network resource-allocation tasks which are divided into independent sub-tasks to be performed in parallel. To this end, not only the individual needs of joining end devices, but also the global network status and existing constraints, are balanced for end-device allocation and provisioning. The functionality of the agents is detailed as follows, with the complexity of the event-based algorithms performed by each of them upon device join being specified:

- *Device Registration*. This agent feeds the MAS with data received via uplink from devices. Having verified that the device concerned is joining the network for the first time, it stores its device EUI and timestamp. The LoRaWAN frame received is then split into *frame payload*, which is forwarded through a socket-based connection to the *Payload Formatting* agent, and metadata, which is forwarded simultaneously to the *Data Rate Discarding* agent via socket for the application of reliability-oriented criteria. The complexity of this agent is $\mathcal{O}(1)$.
- *Payload Formatting*. This agent is responsible for frame payload decoding tasks, which should include, at least, the node's clock skew (T_{sk}), uplink period (T_{ul}), and information about the active sensors. This information is received from the *Device Registration* agent upon device join and used to compute the byte size of future data packets by accessing a magnitude dictionary that will provide the minimum number of payload bits required for each standard magnitude. Eventually, decoded data are adapted to the agent network data format (if any), forwarded to *Time-on-Air Calculation* and *Resource Allocation* agents via socket, and stored in the global device database. The nature of this agent is, therefore, purely reactive, the main objective of which is to provide the required processed data fields as soon as possible for time on air estimation in subsequent stages. The complexity of this agent is $\mathcal{O}(m \times n)$ where m is the payload length of the requesting node in bits and n is the number of entities in the magnitude dictionary.
- *Data Rate Discarding*. This agent establishes, based on RSSI and SNR, a set of reliability-oriented priorities that will determine the SFs' suitability for the joining device. Thus, it provides *Resource-Allocation* agent with a set of mandatory requirements (forbidden SFs based on link quality) and recommendations (preferred SFs). In the case of devices implementing Aloha during low network loads, time-on-air equalization is applied to minimize collisions due to overlapping transmissions using the same bandwidth and

SF, which has already been validated in the literature for minimizing the chances of overlapping transmissions in frequency and time [6]. The complexity of this agent is $\mathcal{O}(n)$ where n is the number of simultaneous schedules at gateway level, that is, one per SF.

- *Time-on-Air Calculation*. As soon as *Data Rate Discarding* and *Payload Formatting* agents provide the desired payload length and prioritized SFs, respectively, Semtech's equations [12] are used by this agent to compute the associated time on airs in parallel. The resulting time-on-air array is automatically forwarded to the *Resource Allocation* agent which will acknowledge upon reception and make a final decision based on the current network status and availability of resources decoupled and provided by previous agents via socket connection. The complexity of this agent is $\mathcal{O}(n)$ where n is the number of simultaneous schedules at gateway level, that is, one per SF.
- *Resource Allocation*. This agent considers specific node constraints to assign it to an existing – or newly-created – NSSE instance based on the network's resource availability. To do so, channel occupancies and downlink usages associated with the joining device are computed for each of the feasible SFs (already filtered in previous stages through sequential decoupling and analysis of request payloads from end devices). These results are combined with the node's SF priorities (provided by *Data Rate Discarding* agent based on the channel-access strategy being applied at the moment). At this point, hardware and application-specific constraints are no longer relevant, since the impact of any node joining the network will be quantitatively assessed. This agent is responsible for short-term decision-making concerning: (i) the SF over which to transmit, (ii) the slot structure to be implemented at the NSSE side, and (iii) the synchronization period to apply. This is, therefore, a key agent in the system, the design criteria for decision-making and complexity of which are detailed in Section 4.4.
- *Instance Deployer*. This agent is purely reactive, and is responsible for launching or terminating NSSE instances on demand according to decisions made by consensus by all agents. Its complexity is $\mathcal{O}(1)$.

4.3. Implementation details

The development of the multi-agent network management module was carried out using Python 3.6.9, following a distributed approach to enable a further cloud-based deployment using Python Microservices. Each agent represents the minimal functional unit in the system, which encompasses a set of pre-defined tasks, objectives, and communication interfaces to enable information exchange. As the system's control logic is distributed, resource allocation sub-tasks are isolated from one another and can be independently updated, scaled, or replaced.

Agents run in a single virtual machine (VM), with socket-based IPC being used for agent-oriented message exchange. Specifically, TCP sockets were used for agent-to-agent communication, since the proposed architecture can leverage pre-established communication flows on initialization, thus achieving greater reliability than that of using UDP sockets. Interconnections between agents are based on bi-directional data streams, while the *frame forwarding* component acts simultaneously as TCP client (network management and data processing) and MQTT subscriber (gateway-level synchronization).

Information storage and management is done through two PostgreSQL databases: *Sensor Data* and *Network Management*. While the former is intended for storing application-specific data, the latter includes two tables that are accessible by all agents to update or retrieve context information upon device discovery. The first table, *Discovery*, includes all the relevant data fields to allocate device-level resources (timestamp, device EUI, RSSI, SNR, data payload, time on air, control payload, uplink period, synchronization period, clock skew, slot length, and roaming ID). The second, *Statistics*, includes cumulative network metrics used for network-status monitoring, such as channel occupancies and downlink usages.

4.4. Resource allocation workflow

The proposed methodology for resource allocation is built upon two improvement stages: *slot-length computation* (definition of slot structures for each NSSE instance according to the available information gathered from the network), and *end-device allocation* (according to the most convenient NSSE instance). Decision-making in these two stages is performed by the *Resource Allocation* agent based on the information, requirements, and recommendations provided by the other agents in the network.

Since the computation of slot lengths depends on previous knowledge, an additional *warm-up* stage is required so as to gather information from the network during low traffic loads. In these cases, there is no need to apply time-slotted scheduling, since LoRaWAN legacy will perform similarly in terms of reliability due to low channel occupation ratios. As a result, the proposed resource-allocation methodology consists of the following stages (detailed in Fig. 5):

(i) *Warm-up period*. The agent will collect data from new end devices joining the network until a certain threshold, based on the overall channel occupancy, is overcome. Below such a threshold, no downlink configuration will be provided, so that an Aloha schema will be followed guaranteeing low overhead.

(ii) *Slot-length computation*. Once the threshold is overcome, slot lengths are computed based on Eq. (4) for each of the schedules that will run in parallel (one per each SF and channel in single-gateway networks). The guard time of each instance is defined according to different criteria (specified in Section 5.2) by using Eq. (5). The complexity of this stage is $\mathcal{O}(n^2)$ where n is the number of simultaneous schedules at gateway level, that is, one per SF.

(iii) *End-device allocation*. Not to be confused with LoRaWAN's legacy join request, in this stage the requester's time on air is computed by using Eq. (6), according to payload-length requirements decoded by *Payload Formatting* agent. If the time on air is equal to or lower than the baseline time on air ($T_{tx_{0_i}}$ in Fig. 5), the required synchronization period and guard time will be automatically calculated. Otherwise, more than one slot will be considered if downlink duty cycles result higher than a certain limit, since an excessively short guard time would congest downlink communication at gateway level. Having defined guard times and synchronization periods for each SF, the uplink channel occupancies and their associated downlink usages are computed, which will serve as a reference to select the most suitable SF for the end device at hand. To do this, the *Resource Allocation* agent will apply different decision-making criteria (specified in Section 5) based on the current network load. The complexity of this stage is $\mathcal{O}(m \times n)$ where m is the number of consecutively-occupied slots at a specific schedule (in a worst-case scenario, equal to the total number of time slots in a synchronization period) and n is the number of simultaneous schedules at gateway level, that is, one per SF.

5. Results and discussion

A set of experiments was carried out to validate the resource-allocation system by measuring the achieved improvements in network capacity, with two main scenarios being tested: (i) a *baseline* scenario where no agents were involved in network setup and management tasks, and (ii) an *agent-based* scenario, where the proposed intermediary system negotiates slot lengths upon launching scheduling and allocates devices on demand according to the network status. In order to provide a fair comparison, the end devices and NSSE, in both cases, implement LoRaWAN-based TDMA over a single gateway and uplink sub-band.

Network monitoring metrics were collected during the experiments for each of the two scenarios being tested until the gateway was unable

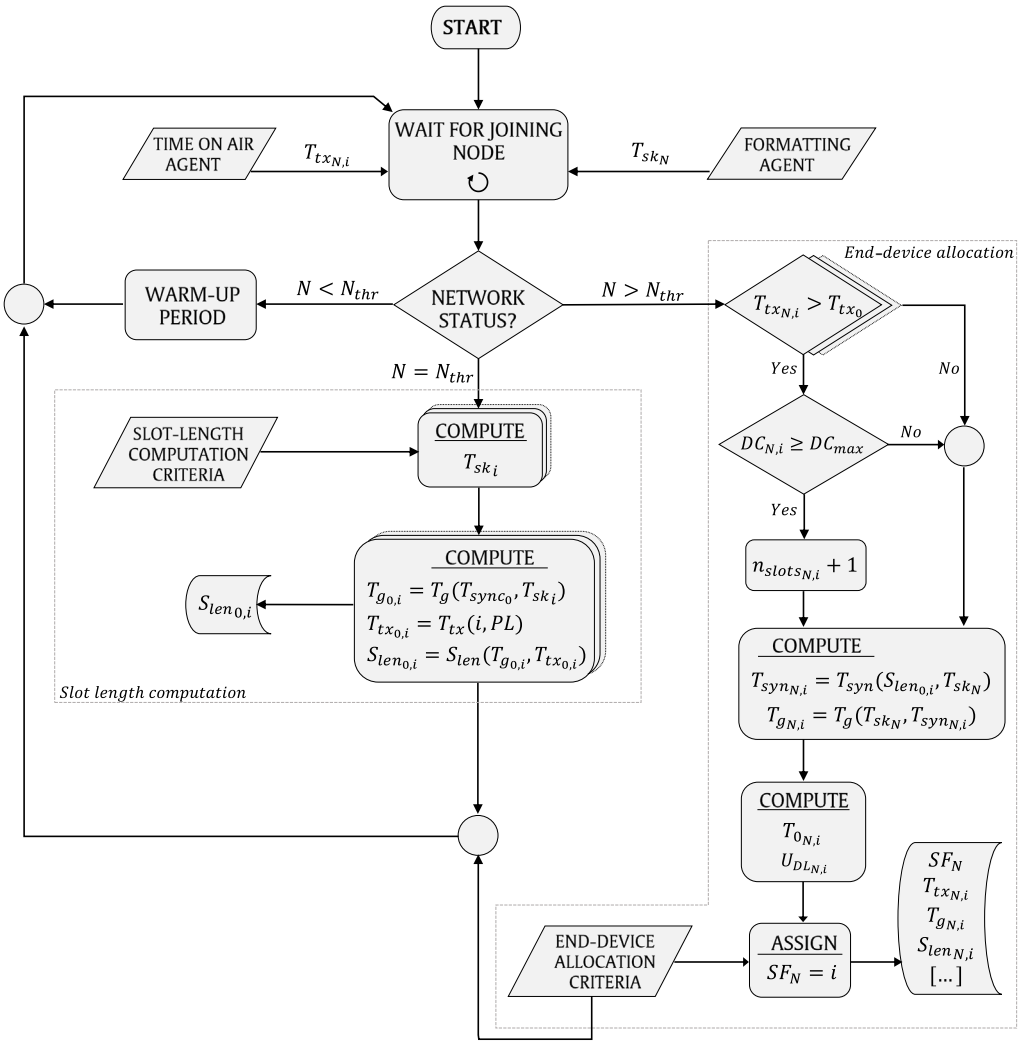


Fig. 5. Flow chart of decision-making performed at the *resource-allocation* agent.

to allocate further end devices. A set of decision-making criteria was tested during *slot-length computation* and *end-device allocation* stages to evaluate the impact of different resource-allocation strategies.

Having validated the LoRaWAN scheduling and synchronization algorithm on top of real hardware in previous works, a LoRaWAN device emulator was selected in the current work to generate high loads of traffic in the network. This component sits before the LoRaWAN network server, thus simultaneously playing the role of *end device(s)*, *gateway(s)*, and *gateway bridge*. The LoRaWAN frames received at the network server are, nevertheless, identical to those that are sent by real end devices. To this end, the ChirpStack Simulator [38], which provides an open-source solution to set up a configurable number of devices and gateways, was used. Some slight modifications were required to enable customizable payload contents, data-reporting intervals, gateway-associated metadata such as SNR or RSSI, and configurable input SF distributions.

5.1. System setup

The experimental setup is shown in Tables 2 and 3, which, respectively, detail the configured parameters for the proposed multi-agent module and ChirpStack's device emulator.

In order to avoid early network congestion due to most devices being assigned to the same schedule, a maximum occupation threshold was defined (see $T_{0_{thr}}$ in Table 2). As a result, when a specific schedule reaches that threshold, the remaining slots are reserved so as not to limit resynchronization capabilities at other concurrent schedules (downlink). With regard to duty cycling, agents compute slot lengths, ensuring that no device exceeds a certain downlink communication

Table 2
Multi-agent network management module configuration.

Parameter	Description	Value
T_{syn_0}	Reference synchronization period	120 min
T_{sk_0}	Reference theoretical clock skew	20 ppms
N_{thr}	Threshold to launch scheduling	100 devices
$T_{0_{thr}}$	Threshold to disable SF allocation	80%
DC_{max}	Maximum downlink duty cycle	0.05%

Table 3
Setup parameters for devices' emulation.

Parameter	Description	Value
t	Emulation time	8000 s
N	Number of devices	2500 devices
N_{SF_A}	SF7-12 urban weighted probability	[0.3, 0.2, 0.2, 0.1, 0.1, 0.1]
N_{SF_B}	SF7-12 rural weighted probability	[0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
T_{sk}	Linear distribution of clock skews	[5, 75] ppms
T_{UL}	Linear distribution of tx period	[5, 30] min
PL_{FRM}	Frame payload	3 bytes
PL_{PHY}	PHY payload	16 bytes

threshold (DC_{max} in Table 2) to avoid network congestion, in this case, due to excessively-frequent resynchronizations.

As shown in Table 3, devices joining the network for the first time base their requesting SF on a weighted random probability, with SF distributions A and B being derived from a recent coverage study conducted in the city of Albacete (Spain) for rural and urban scenarios [39]. The rural distribution, for instance, consists of a 30% probability of using SF7, 20% of using SF8 to SF9, and 10% of using SF10 to SF12. Considering a uniform geographical distribution of end devices around a gateway with SF7 being used for distances smaller than 1.5 km (in line with the results in [39]), this is translated into baseline distances of up to 1.94 km for SF8, 2.74 km for SF9, 3.35 km for SF10, 4.73 km for SF11 and 6.68 km for SF12. The clock skew and uplink period of joining devices follow a linear distribution (intervals specified in Table 3) with their payload length being constant to focus on guard-time computation improvements. Clock skew, specifically, was set up in the range of 5 to 75 ppms to cover representative cases identified in the experimental results obtained in [10]. Finally, Cayenne Low-Power Payloads[1] were considered for bit-wise payload formatting, the use of which enabled temperature and humidity encoding with 0.1 °C and 0.5% resolution, respectively.

5.2. Testing scenarios and decision-making criteria

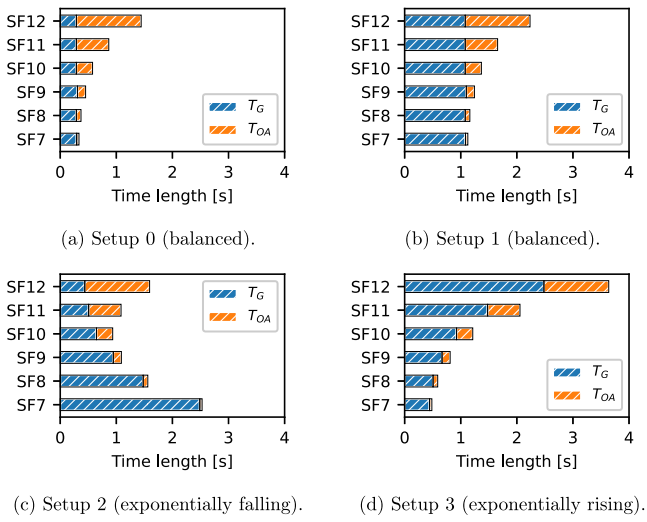
Slot-length computation criteria are based on the information gathered by agents over time (please, refer to Fig. 5). These provide a significant improvement with respect to offline scheduling and synchronization, which compute the required slot lengths in the network considering the collected metadata. The following setups are proposed for testing:

- *Setup 1* (balanced). The maximum skew was used as a reference to compute the guard time for all SFs, with all of these being assigned the same guard time length.
- *Setup 2* (exponentially falling). The assigned guard times per schedule follow an exponentially-falling distribution, with their sum being equal to the sum of guard times in *Setup 1* in order to enable a fair comparison.
- *Setup 3* (exponentially rising). Similar to the previous setup, with the assigned guard times per SF schedule following an exponentially-falling distribution.

Having computed slot lengths, the following strategies are considered for *end-device allocation* (refer to Fig. 5):

- *Strategy 1* (lowest SF). By considering the maximum occupation threshold (see Table 2), agents allocate the joining end node to the lowest-possible SF following a pseudo-ADR schema below occupation thresholds.
- *Strategy 2* (lowest resources). Agents compute uplink channel occupancies and downlink usages for each SF (Eqs. (7) and (8)) and assign the node to the SF requiring the lowest use of global – uplink and downlink – resources.
- *Strategy 3* (network-aware resources). Based on the cumulative channel occupancies and downlink usages, Eqs. (7) and (8) are applied to quantify the impact of allocating the end device at hand to each of the available SFs. If uplink resources are closer to congestion than downlink ones, the SF that results in the lowest uplink occupancy will be assigned. Otherwise, the SF with the lowest-possible downlink usage will be allocated. This strategy is aimed at equalizing uplink and downlink resources, following a similar approach to EXPLoRa-AT [6], but considering the overall network resources.

[1] https://docs.mydevices.com/docs/lorawan/cayenne-lpp.



(a) Setup 0 (balanced). (b) Setup 1 (balanced).

(c) Setup 2 (exponentially falling). (d) Setup 3 (exponentially rising).

Fig. 6. Slot lengths upon scheduling launching for *setups 0* to *3* (T_G stands for *guard time* and T_{OA} stands for *time on air*).

These resource-allocation strategies provide the required capacity improvements in the network, not only by dynamically assigning synchronization periods to new joining devices according to their clock skew, but also by delaying network congestion based on (uplink and downlink) channel utilization. The proposed agent-based setups and strategies were tested for the two SF distributions of Table 3 and compared in terms of scalability with a baseline scenario lacking of agents. This baseline scenario consists of:

- *Setup 0*. A typical clock skew (see Table 2) is considered to compute guard times for all the SFs, since no previous knowledge is available.
- *Strategy 0*. LoRaWAN's ADR legacy strategy is used to allocate each device to a schedule, thereby assigning the minimum possible SF in terms of signal quality.

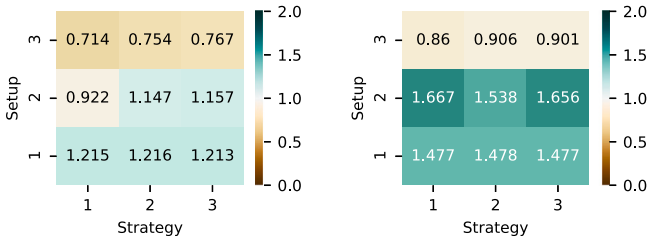
5.3. Scalability improvements

Fig. 6 shows the resulting slot lengths after guard-time computation for each of the tested setups, with the *baseline* being shown in Fig. 6(a) and the *agent-based* in Figs. 6(b) to 6(d). Note that, in the latter, the sum of slot lengths using SF7 to SF12 remains constant for all the setups.

An overall comparison of the system's performance in terms of network capacity is shown in Fig. 7 for different combinations of *agent-based* setups (see Fig. 6) and allocation strategies, with the achieved network sizes being normalized with respect to the *baseline* scenario. At first sight, it is clear that the geographical distribution of end nodes determines the most convenient strategy and setup to apply to extend the network's scalability. Nevertheless, it is the latter that achieves significant improvements in terms of maximum achievable network size.

Based on the normalized results from Fig. 7, a set of representative *setup-strategy* pairs were selected for further analysis at both *slot-length computation* and *device allocation* stages. These are represented with the setup identifier followed by the strategy identifier, e.g. *SE1_ST1*.

Fig. 8 shows channel occupancies and downlink usages as the network size increases, with end nodes being deployed using geographical distribution A. The vertical dotted lines represent network congestion, which may be due to either: (i) downlink congestion with usage close to 100% (BPs included), or (ii) uplink congestion with a 100% channel occupancy in any of the available SF schedules. The overall channel occupancy is computed considering the sum of the six concurrent occupancies due to SF orthogonality.



(a) Geographical distribution A. (b) Geographical distribution B.

Fig. 7. Normalized network size for each setup-strategy pair.

Some preliminary conclusions can be drawn from Fig. 8. The longer the guard time (e.g. *SE3_ST3*), the lower is the number of available uplink slots per time unit. Hence, channel occupancies increase at a higher rate than those of setups with shorter guard times, leading to uplink congestion in at least one of the schedules. As it can be seen, additionally, when the guard time is considerably short (e.g. *SE0_ST0*), the more frequent resynchronizations generate early network congestion due to significantly high downlink usages. Therefore, it is key for agents to balance uplink and downlink in order to extend the network's scalability, which mostly depends on the criteria applied during *slot-length computation* (see Fig. 7).

Fig. 9 shows the resulting distribution of nodes per SF schedule after decision-making. Note that, in the *baseline* scenario (*SE0_ST0*), the resulting distribution of end nodes per SF corresponds exactly to the input distribution shown in Table 3, since the lowest-possible SF is automatically assigned. In the remaining cases, the selected setup and strategy will determine the most convenient schedule for each of the requesting nodes. As a result, a certain number of nodes are transferred by agents from lower to higher SFs, with the impact of such allocation being shown in Fig. 8.

The remaining scenarios need further analysis since it was the criteria adopted during the *end-device allocation* stage and the geographical distribution of end nodes that conditioned the network size. For this purpose, representative *agent-based* scenarios are shown in Fig. 10, where a breakdown of occupancies per SF and their impact on downlink is provided.

When *Setup 1*'s slots were used, agents assigned joining nodes to the lowest possible SF regardless of the allocation strategy applied, since slot lengths increased as the SF did so. This can be seen in Fig. 9(b), where the resulting allocation of end nodes was identical for *SE0_ST0* and *SE1_ST2* when these followed geographical distribution B. Nevertheless, when the nodes followed distribution A, certain benefits were obtained from disabling SF allocation upon threshold overcoming ($T_{0_{thr}}$ in Table 2) which, as shown in Fig. 9(a) (*SE1_ST2*), forced the system to allocate joining nodes to higher SFs when lower ones were overloaded.

Such benefits can be seen in Figs. 10(a) and 10(b) where, given a higher proportion of devices joining with lower SFs, individual schedules overcame their thresholds and forced agents to assign incoming devices to higher SFs. As a result, channel occupancies per SF were, to some extent, balanced, which delayed network congestion in comparison to other scenarios. This scenario achieved the largest network size when nodes followed distribution A, with improvements of 21.60% compared to the *baseline* being achieved.

In Fig. 10(c), given the relatively long guard time reserved for low SFs (which can be seen in Fig. 6) and an allocation strategy that consisted of assigning the lowest possible SF, uplink channel occupancies rose rapidly until network congestion. Similarly as in the previous scenario, individual SFs overcame allocation thresholds but, in this case, did not prevent the network from uplink congestion due to excessively long slot lengths at lower SFs and the high proportion of nodes located close to the gateway under distribution A. In addition, the allocation of higher SFs caused a more significant impact on downlink usage (see Fig. 10(d)) due to more frequent resynchronizations. Fig. 8 shows how, given an early SF7 congestion, the network left around 25% (uplink) and 50% (downlink) of resources unused.

In contrast, the same scenario (*SE2_ST1*) achieved the best results when distribution B was tested. This can be seen in Figs. 10(g) and 10(h), where a larger proportion of nodes assigned to higher SFs resulted in a more balanced uplink-to-downlink ratio as well as load among the different uplink schedules. In this case, network-size improvements were up to 66.70% compared to these of the *baseline* scenario when the nodes followed distribution B.

With regard to the *SE2_ST2* scenario (Figs. 10(e) and 10(f)), since guard times decreased with increasing transmission times, agents tended to transfer nodes from lower SFs to SF9-10, which implied a trade-off between uplink and downlink use of resources. Hence, when possible, the network avoided the shortest guard times that would imply frequent resynchronizations (e.g. SF12) as well as the longest slot lengths that might rapidly congest individual SFs (as happened in the scenario shown in Fig. 10(c)).

As it can be observed in Fig. 9(a), only around 5% of devices were assigned to SF7, while the remaining 95% were balanced accross SF8 to SF12. This is due to the allocation strategy, which first prioritized SF9 and SF10 (depending on the nodes' clock skew), then SF8 and SF11 and, eventually, SF7 and SF12. However, when SF8 to SF11 reached their thresholds (see Fig. 10(e)), any node joining with an SF higher than SF7 was necessarily assigned to SF12, given their RSSI constraints. This behaviour can also be observed in Fig. 10(e), where occupancies rose simultaneously for each pair of SFs and, in the case of SF7 and SF12, the differences in slot lengths determined the rate of rising. The achieved network size improvements, in this case, were 15.72% with respect to distribution A.

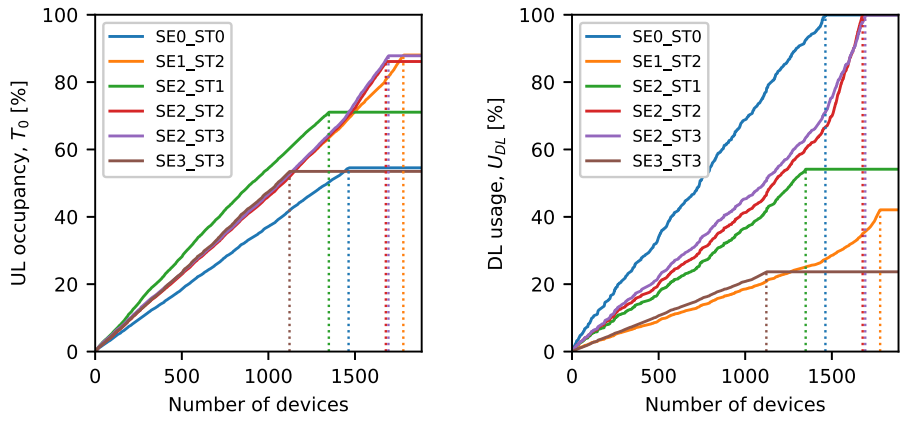
Eventually, *Setup 3* provided no significant improvements compared to *baseline* network, even under a network-aware strategy (*Strategy 3*). As seen in 9(b), agents transferred devices to SF9 when possible since, based on the measured network status, the use of uplink resources needed to be minimized. Despite assigning lower SFs – which would imply a lower uplink occupancy based on their shorter slot length – agents tended to assign SF9. This is due to the impact of slot reservation, as the shortest guard time also results in more frequent synchronizations and, hence, higher reservation ratios.

Figs. 10(i) and 10(j) show how, despite targeting the reduction of uplink resources, longer slot lengths at higher SFs increased rapidly their channel occupancies. This is not only due to nodes joining at such SFs, but also to frequent resynchronizations at lower SFs (relatively short guard times) and the impact of slot reservation. The remaining strategies for *Setup 3* yielded similar results, as shown in Fig. 7.

5.4. Implications in real-world deployments

The results presented in this work were obtained following a hybrid approach in which end nodes joining the network were emulated and the multi-agent network management system proposed was implemented and validated experimentally. Moreover, an end-to-end real-world deployment was carried out using a reduced set of STM32L072Z micro-controller units (MCUs) to validate device-to-MAS, MAS-to-NSSE, MAS-to-device, and device-to-NSSE communication links. Nevertheless, there exist different bottlenecks derived from larger-scale network deployments in practice, which are briefly addressed in this section to propose potential solutions to tackle them.

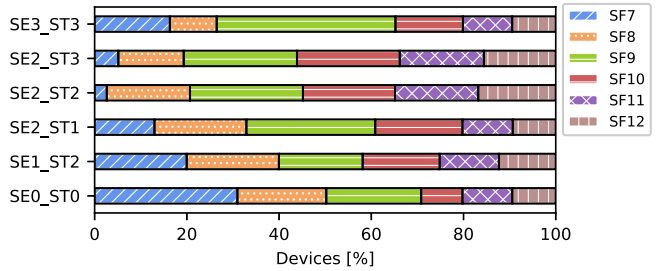
Despite the coexistence of heterogeneous clock sources in the same network being addressed in this work, each node is required to know its own clock skew in advance. This, in turn, would require the development of a semi-automated method for online clock skew measurement instead of relying on the theoretical specifications provided by the device manufacturer so as to achieve optimal results. This could be addressed by leveraging downlink signalling provided by the NSSE



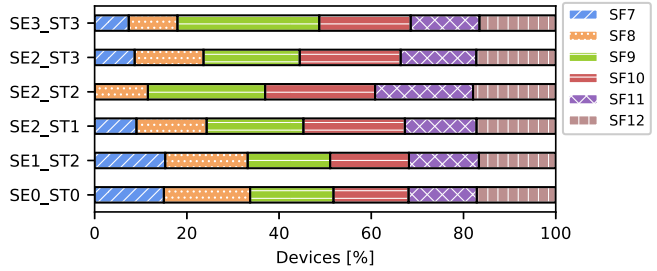
(a) Channel occupancy over time.

(b) Downlink usage over time.

Fig. 8. Network metrics for different slot-length computation strategies and LoRaWAN's legacy end-device allocation (geographical distribution A). *SE* stands for *Setup* and *ST* stands for *strategy* according to definitions provided in Section 5.2.



(a) Geographical distribution A.



(b) Geographical distribution B.

Fig. 9. Per-schedule distribution of end devices according to different *slot-length computation* and *end-device allocation* criteria.

between two consecutive synchronization requests so that the node provides a theoretical clock skew value upon joining and refines it over time based on the time references provided by the NSSE.

The instability of RSSI and SNR is another bottleneck arising in real-world deployments, since part of the decision-making taking place at the MAS-side relies on these measurements. To overcome this issue, a real-world dataset concerning different types of scenarios (e.g. industrial, urban or rural) could be generated and leveraged to train a machine learning (ML) prediction model based on open-source information from traffic or weather forecasts, among others.

Regarding the end-node distribution with respect to the gateway (geographical distributions A and B assessed in this work), a set of lessons learned are provided to support improvements in the network capacity according to the expected node distribution required by the application being pursued.

6. Conclusions

This work addressed the design and implementation of a multi-agent approach to resource allocation in LoRaWAN networks, tested with a set of emulated end nodes in order to measure the achieved scalability improvements. The proposed system was built on top of LoRaWAN's MAC layer, which consists of a scheduling algorithm to be implemented at both ends – LoRaWAN devices and application server – and a set of agent-based network resource management components. With the focus placed on the latter, this work provides a detailed architecture design, system logic, communication flows, and decision-making criteria to be implemented by individual agents. These collaborate to compute channel utilization statistics online and, hence, dynamically assign nodes to the most convenient schedule leveraging TDMA transmissions and SF orthogonality in single-gateway networks.

6.1. Contributions and findings

The proposed system acts as a network management abstraction layer, which processes join requests on demand and negotiates with existing NSSEs the allocation of devices based on their physical constraints, communication requirements, and the current network status. As part of the experimental setup, the suitability of agent network components was validated by quantifying the impact of different decision-making criteria on the maximum network capacity achieved. To this end, a set of agent configuration scenarios were proposed and tested through ChirpStack's device emulator, which mainly focused on two stages: (i) *slot length computation*, and (ii) *end-device allocation*. The results obtained yield interesting findings related to decision-making in time-slotted LoRaWAN networks, which represent a significant contribution to the existing literature and are summarized as follows:

(i) The geographical distribution of end nodes is a determinant for selecting the most convenient guard-time distributions per SF in the network (*slot length computation* stage) as well as the *end-node allocation strategy* to be followed by resource-allocation agents. This highlights the importance of conducting online resource allocation which, to date, has received scant attention in the literature. Significant differences were found in the network capacity when applying different slot-length computation strategies in urban-like and rural-like deployment distributions.

(ii) The overall uplink occupancies should be balanced with downlink usages based on the number of available gateways and their location with respect to end nodes. Once a trade-off is reached, balancing uplink occupancies across different SFs – scheduled channels – can delay network congestion by exploiting orthogonality and network-wide knowledge at the network management side.

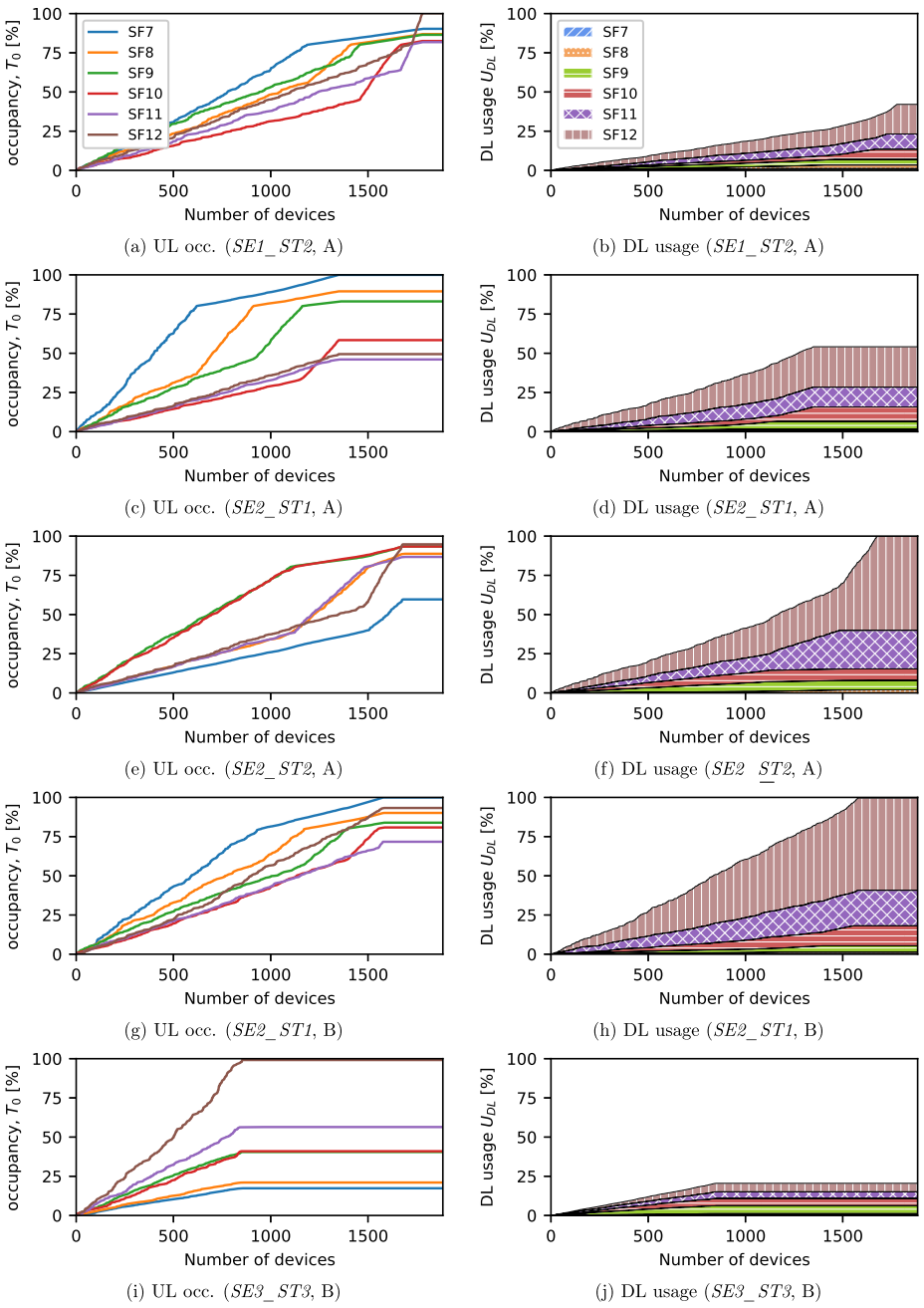


Fig. 10. SF breakdown of cumulative occupancies (T_0) and downlink usages (U_{DL}) in the network under LoRaWAN's legacy end-device allocation strategy.

(iii) Given duty cycling constraints at the gateway side, *slot-length computation* can emerge as a key stage while performing online scheduling. Guard-time lengths, as well as distributions per schedule, should be carefully selected in order to avoid immediate uplink or downlink congestion due to slot reservation, which will depend on the information gathered by agents during the *warm-up* stage. Hence, it is essential for the network to be able to adjust scheduling thresholds dynamically based on the current network status and physical constraints.

6.2. Implications in practice

The integration of multi-agent components in the system can be conceived as an assessment tool for large-scale network deployments. By following a modular design, the proposed system can be configured for varying network constraints and scenarios based on the end application being pursued, while the results provided in this work serve as a set of lessons learned in the context of LoRaWAN time-slotted scheduling.

On the whole, a novel approach to online resource allocation in time-slotted LoRaWAN networks is presented and validated in this work. First, we leverage a TDMA schema for collision avoidance in multi-SF networks while considering factors influencing the overall achievable network size such as channel utilization or downlink resource sharing, which are typically neglected in reliability-oriented studies given the need for single-SF validation in practice. Second, we propose and validate with the benchmark MAS deployed different SF allocation strategies with a view to achieving network-size improvements through efficient use of resources, while still complying with duty cycle constraints and TDMA design considerations. With this hybrid approach, we achieve network size improvements of up to a 66.7% and, more interesting, we identify different real-world factors that need to be considered for decision-making during resource allocation

such as the geographical distribution of nodes within the gateway cell or the guard-time distribution across parallel SF schedules. The implications of the presented approach for real-world deployments are also discussed, which call for the design of a semi-automated clock skew measurement methodology or a ML model for accurate RSSI and SNR estimation based on open data sources.

6.3. Future work

In future work, we plan to carry out a performance evaluation of the system in terms of agent-based communication latencies and computational load in order to conduct a further microservice-based deployment. Moreover, we aim to add a context management component in the system to make network-wide decisions concerning different co-located gateways. As a result, the system will be extended to minimize the impact of mobile end-nodes, which are an essential aspect of LoRaWAN's scope with growing application domains such as supply chain management.

In addition, the development of a more flexible mechanism to determine agent-based thresholds is also highly desirable, which, based on the results provided in this work, can be used to apply machine learning so as to provide agents with different priorities or goals in advance depending on the network status and forecast information. To do so, a sensitivity analysis of the end-to-end system according to super-parameter settings is proposed.

CRediT authorship contribution statement

Celia Garrido-Hidalgo: Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing – original draft, Approved the version of the manuscript to be published. **Luis Roda-Sanchez:** Acquisition of data, Analysis and/or interpretation of data, Writing – original draft, Approved the version of the manuscript to be published. **F. Javier Ramírez:** Analysis and/or interpretation of data, Writing – review & editing, Approved the version of the manuscript to be published. **Antonio Fernández-Caballero:** Conception and design of study, Writing – review & editing, Approved the version of the manuscript to be published. **Teresa Olivares:** Analysis and/or interpretation of data, Writing – review & editing, Approved the version of the manuscript to be published.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

Grant 2021-GRIN-31042 funded by Universidad de Castilla-La Mancha, Spain. Grant 2019-PREDUCLM-10703 funded by Universidad de Castilla-La Mancha, Spain and by "ESF, Spain Investing in your future". Grant DIN2018-010177 funded by MCIN/AEI/10.13039/5011000 11033. Grant PID2020-117398GB-I00 funded by MCIN/AEI/10.13039/501100011033. Grant PID2021-123627OB-C52 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way to make Europe".

References

[1] K.L. Lueth, State of the IoT & Short-term Outlook-Q1/Q2 2018, Tech. Rep., IoT Analytics GmbH, Hamburg, Germany, 2018.
[2] S. Böcker, C. Arendt, P. Jörke, C. Wietfeld, LPWAN in the context of 5G: Capability of LoRaWAN to contribute to mMTC, in: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE, 2019, pp. 737–742.
[3] A. Čolaković, M. Hadžialić, Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues, Comput. Netw. 144 (2018) 17–39.
[4] G. Sun, K. Xiong, G.O. Boateng, D. Ayepah-Mensah, G. Liu, W. Jiang, Autonomous resource provisioning and resource customization for mixed traffics in virtualized radio access network, IEEE Syst. J. 13 (3) (2019) 2454–2465.
[5] G. Sun, D. Ayepah-Mensah, R. Xu, V.K. Agbesi, G. Liu, W. Jiang, Transfer learning for autonomous cell activation based on relational reinforcement learning with adaptive reward, IEEE Syst. J. 16 (1) (2021) 1044–1055.
[6] F. Cuomo, M. Campo, A. Caponi, G. Bianchi, G. Rossini, P. Pisani, EXPLoRa: Extending the performance of LoRa by suitable spreading factor allocations, in: 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), IEEE, 2017, pp. 1–8.
[7] A. Duda, M. Heusse, Spatial issues in modeling LoRaWAN capacity, in: Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2019, pp. 191–198.
[8] D. Zorbas, Design considerations for time-slotted LoRa(WAN), in: MaDeLoRa 2020: 1st Workshop on Massive LoRa Deployments: Challenges and Solutions, 2020.
[9] J. Haxhibeqiri, I. Moerman, J. Hoebeke, Low overhead scheduling of LoRa transmissions for improved scalability, IEEE Internet Things J. 6 (2) (2018) 3097–3109.
[10] C. Garrido-Hidalgo, J. Haxhibeqiri, B. Moons, J. Hoebeke, T. Olivares, F.J. Ramirez, A. Fernández-Caballero, LoRaWAN scheduling: From concept to implementation, IEEE Internet Things J. (2021).
[11] LoRaWAN^{TM} 1.0.3 Specification, 2018, Rev. 1.0.3.
[12] SX1272/3/6/7/8: LoRa modem, 2013, Rev. 1.
[13] C. Caillouet, M. Heusse, F. Rousseau, Optimal SF allocation in LoRaWAN considering physical capture and imperfect orthogonality, in: 2019 IEEE Global Communications Conference (GLOBECOM), IEEE, 2019, pp. 1–6.
[14] I. Tomić, L. Bhatia, M.J. Breza, J.A. McCann, The limits of LoRaWAN in event-triggered wireless networked control systems, in: 2018 UKACC 12th International Conference on Control (CONTROL), IEEE, 2018, pp. 101–106.
[15] J. Moraes, H. Oliveira, E. Cerqueira, C. Both, S. Zeadally, D. Rosário, Evaluation of an adaptive resource allocation for LoRaWAN, J. Signal Process. Syst. (2021) 1–15.
[16] R. Kufakunesu, G.P. Hancke, A.M. Abu-Mahfouz, A survey on adaptive data rate optimization in LoRaWAN: Recent solutions and major challenges, Sensors 20 (18) (2020) 5044.
[17] A. Loubany, S. Lahoud, R. El Chall, Adaptive algorithm for spreading factor selection in LoRaWAN networks with multiple gateways, Comput. Netw. 182 (2020) 107491.
[18] K.Q. Abdelfadeel, V. Cionca, D. Pesch, Fair adaptive data rate allocation and power control in LoRaWAN, in: 2018 IEEE 19th International Symposium on" a World of Wireless, Mobile and Multimedia Networks"(WoWMoM), IEEE, 2018, pp. 14–15.
[19] A. Tellache, A. Mekrache, A. Bradai, R. Boussaha, Y. Pousset, Deep reinforcement learning based resource allocation in dense sliced LoRaWAN networks, in: 2022 IEEE International Conference on Consumer Electronics (ICCE), IEEE, 2022, pp. 1–6.
[20] G. Park, W. Lee, I. Joe, Network resource optimization with reinforcement learning for low power wide area networks, EURASIP J. Wireless Commun. Networking 2020 (1) (2020) 1–20.
[21] M. Hamnache, R. Kacimi, A.-L. Beylot, L3SFA: Load shifting strategy for spreading factor allocation in LoRaWAN systems, 2020.
[22] E. Sallum, N. Pereira, M. Alves, M. Santos, Improving quality-of-service in LoRa low-power wide-area networks through optimized radio resource management, J. Sens. Actuator Netw. 9 (1) (2020) 10.
[23] E. Lima, J. Moraes, H. Oliveira, E. Cerqueira, S. Zeadally, D. Rosário, Adaptive priority-aware LoRaWAN resource allocation for Internet of Things applications, Ad Hoc Netw. 122 (2021) 102598.
[24] J. Haxhibeqiri, E. De Poorter, I. Moerman, J. Hoebeke, A survey of LoRaWAN for IoT: From technology to application, Sensors 18 (11) (2018) 3995.
[25] H. Rajab, T. Cinkler, T. Bouguera, IoT scheduling for higher throughput and lower transmission power, Wirel. Netw. 27 (3) (2021) 1701–1714.
[26] A. Triantafyllou, P. Sarigiannidis, T. Lagkas, I.D. Moscholios, A. Sarigiannidis, Leveraging fairness in LoRaWAN: A novel scheduling scheme for collision avoidance, Comput. Netw. 186 (2021) 107735.
[27] L. Chasserat, Achieving energy efficiency in dense LoRaWANs through TDMA, in: IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE, 2020.
[28] K.Q. Abdelfadeel, D. Zorbas, V. Cionca, D. Pesch, *FREE*—Fine-grained scheduling for reliable and energy-efficient data collection in LoRaWAN, IEEE Internet Things J. 7 (1) (2019) 669–683.

[29] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, S. Pollin, Improving reliability and scalability of lorawans through lightweight scheduling, IEEE Internet Things J. 5 (3) (2018) 1830–1842.
[30] T.-H. To, A. Duda, Timemaps for improving performance of LoRaWAN, in: ICC 2020-2020 IEEE International Conference on Communications (ICC), IEEE, 2020, pp. 1–7.
[31] R. Piyare, A.L. Murphy, M. Magno, L. Benini, On-demand LoRa: Asynchronous TDMA for energy efficient and low latency communication in IoT, Sensors 18 (11) (2018) 3718.
[32] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, D. Pesch, TS-LoRa: Time-slotted LoRaWAN for the industrial Internet of Things, Comput. Commun. 153 (2020) 1–10.
[33] J. Finnegan, R. Farrell, S. Brown, Lightweight timeslot scheduling through periodicity detection for increased scalability of LoRaWAN, in: 2020 IEEE 21st International Symposium on" a World of Wireless, Mobile and Multimedia Networks" (WoWMoM), IEEE, 2020, pp. 8–15.
[34] D. Zorbas, K.Q. Abdelfadeel, V. Cionca, D. Pesch, B. O'Flynn, Offline scheduling algorithms for time-slotted LoRa-based bulk data transmission, in: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE, 2019, pp. 949–954.
[35] A. Triantafyllou, D. Zorbas, P. Sarigiannidis, Time-slotted LoRa MAC with variable payload support, Comput. Commun. 193 (2022) 146–154.
[36] Spectrum occupancy measurements and evaluation, 2018, Rev. 1.
[37] B. Moons, A. Karaagac, E. De Poorter, J. Hoebeke, Efficient vertical handover in heterogeneous low-power wide-area networks, IEEE Internet Things J. 7 (3) (2019) 1960–1973.
[38] ChirpStack, Open source LoRaWAN network server stack, 2021, https://www.chirpstack.io/, Accessed: 15-9-2021.
[39] C. Garrido-Hidalgo, T. Olivares, F.J. Ramirez, L. Roda-Sanchez, An end-to-end internet of things solution for reverse supply chain management in industry 4.0, Comput. Ind. 112 (2019) 103127.

Celia Garrido-Hidalgo received her M.Sc. in Industrial Electronics Engineering in 2018, and her Ph.D. in Advanced Computing Technologies in 2022, both from Universidad de Castilla-La Mancha, Spain. She is currently Postdoctoral researcher at Instituto de Investigación en Informática de Albacete (Spain) from the High-Performance Networks and Architectures group. Celia has received different international awards for innovation projects, and her research interests include low-power communication standards, heterogeneous networks and reverse supply chain management.

Luis Roda-Sanchez received the M.Sc. degree in industrial engineering specialized in electronics in 2018 from the University of Castilla-La Mancha (Spain). He is member of the research group High Performance Networks and Architectures from the Albacete Research Institute of Informatics. He received an Industrial Ph.D. scholarship from Spanish Ministry of Science and Innovation with NEC Ibérica and, since 2022, he is Ph.D. in advanced computing technologies. His principal research interests are Internet of Things, Smart Cities, Industry 4.0, Edge computing, wireless sensor networks, service orchestration and Industrial Internet of Things (IIoT).

F. Javier Ramírez is Tenured Professor of Engineering Management at the School of Industrial Engineering, Universidad de Castilla-La Mancha, Albacete, Spain. Ramirez's research focuses on operations research and manufacturing engineering. He has participated in more than 120 R&D projects and has co-authored over 70 research papers in journals, specialized conferences and book chapters. He is co-inventor of 6 patents with industrial exploitation. F. Javier Ramirez is Associate Editor of Proceedings of the Institution of Mechanical Engineers Part C: Journal of Mechanical Engineering Science, and Cogent Engineering, and Guest Editor of The International Journal of Advanced Manufacturing Technology.

Antonio Fernández-Caballero received his M.Sc. in Computer Science from Universidad Politécnica de Madrid, Spain, in 1993, and his Ph.D. from the Department of Artificial Intelligence at Universidad Nacional de Educación a Distancia, Spain, in 2001. He is a Full Professor with the Department of Computer Science at Universidad de Castilla-La Mancha. He is head of the n&aIS (natural and artificial Interaction Systems) research group belonging to the Albacete Research Institute of Informatics. His research interests are Image Processing, Cognitive Vision, Mobile Robotics, Affective Computing and Intelligent Agents. Antonio Fernández-Caballero is Associate Editor of Pattern Recognition Letters, Topic Editor-in-Chief for Vision Systems of International Journal of Advanced Robotic Systems, and Specialty Chief Editor for Robot and Machine Vision of Frontiers in Robotics and AI, among other editorship tasks. He has authored more than 400 peer-reviewed papers.

Teresa Olivares is Assistant Professor with the Department of Computing Systems at Universidad de Castilla-La Mancha. She received her Ph.D. degree in Computer Science in 2003 from the same university. She is a member of the research group High-Performance Networks and Architectures at the Albacete Research Institute of Informatics. Her main scientific research interests include Internet of Things (IoT) standards, communications and protocols, heterogeneous low power wireless sensor networks and standards, smart environments, Industry 4.0 and Reverse Logistics. She has participated in more than 40 research projects and has co-authored more than 70 research papers in journals, conferences and book chapters.