p-CARMA: Politely Scaling LoRaWAN

Nikolaos Kouvelas[1], Vijay S Rao[2], R. Venkatesha Prasad[1], Gauri Tawde[1], and Koen Langendoen[1]
[1]Embedded and Networked Systems, Delft University of Technology
[2]Cognizant Technology Solutions
{N.Kouvelas, V.Rao, R.R.VenkateshaPrasad, K.G.Langendoen,}@tudelft.nl

Abstract

Long Range Wide Area Network (LoRaWAN) covers the needs of energy-constrained IoT-devices for operational longevity and extended communication range in a best-effort fashion. However, LoRaWAN's minimalist design cannot handle the traffic from dense deployments with more than a few hundred devices connected to a single gateway, since each LoRa-device transmits data-packets without any information regarding the availability of the medium.

In this paper, we try to improve the scalability of LoRaWAN by manifolds, serving thousands of devices per gateway. We present a novel protocol called p persistent-Channel Activity Recognition Multiple Access (p-CARMA) that exploits LoRaWAN's Channel Activity Detection (CAD) as a crude mechanism to assess if the channel is free. Due to CAD's imperfections (it only scans for preambles, not for any channel activity) p-CARMA operates probabilistically with each device deciding on a p value based upon local estimation. At the beginning of operation, this estimate is derived from pure local information, that is without involvement of the gateway, and devices automatically adapt to changes in the environment. Then, the adaptation of p-value is assisted by critical information on the cumulative device-delays, multicasted by the gateway at regular, large timespans.

To evaluate the performance of p-CARMA, we implemented it in *ns*-3 based upon a detailed characterization of LoRaWAN's CAD mechanism involving an extensive set of real-world experiments. We compared p-CARMA to vanilla LoRaWAN as well as a variant using the theoretically optimal $p = 1/N$ (N being the total number of devices). The simulation results show that p-CARMA achieves from three-fold, up to a twenty-fold higher Packet Reception Ratio than LoRaWAN while handling thousands of devices. Further, its adaptivity outperforms the fixed p-value by a factor of 5.25 when scaling up. Moreover, p-CARMA does so while consuming 37.31%-58.17% less energy on average per device compared to vanilla LoRaWAN.

Keywords

LoRaWAN, scalability, carrier sensing, CAD, adaptive p, persistence, hidden terminals

1 Introduction

With the rise and projected scale of the Internet of Things (IoT), many technological solutions have been proposed for various smart (city) applications including smart street lighting, smart meters, smart parking, smart vehicle monitoring, and traffic updates [1]. At the moment, more than 7 billion IoT sensors are connected globally and they are expected to rise to 22 billion by 2025 [2].

Recent advances in RF technology have enabled a simpler technological solution, called Low-Power Wide Area Networks (LPWAN), compared to low-power mesh networking. LPWANs offer low-energy communication over several kilometers, allowing sensors to send data to a central gateway over just a single hop. LoRaWAN (Long Range Wide Area Network) is one of the several competing LPWAN technologies with, among others, SigFox, NB-IoT and Weightless [3]. LoRaWAN has been the most successful of these technologies in providing an easily accessible LPWAN [4]. The protocol is being developed by the open LoRa Alliance [5]. In comparison to other LPWAN technologies, LoRaWAN provides an inexpensive, easily deployable, secure and power-efficient communication method for the class of non-critical IoT applications, which generally send out small payloads at a low rate and are delay tolerant as well as resilient to (some) packet loss.

LoRa is a wireless communication technology based on Chirp Spread Spectrum (CSS), a frequency modulation technique that is robust against channel noise, multi-path fading, and resistant to the Doppler effect [4]. Several measurement campaigns using real-world deployments prove that proper configuration of LoRa-parameters can establish long robust links, even in mobile scenarios [6, 7, 8]. A specification for networking LoRa devices has been defined and is called LoRa Wide Area Networks (LoRaWAN). A star topology is adopted wherein sensor nodes transmit their data to a gateway. We are interested in *class A* of LoRaWAN defined to cater to non-critical IoT applications with several thousands of devices sending frames to a single gateway. To connect that many devices, LoRaWAN provides several configurable transmission parameters – Spreading Factor (SF), transmission power, channel bandwidth, carrier frequency, forward error correction, and coding rate. Of these parameters, SFs manifest pseudo-orthogonality, allowing messages modulated by different SFs (SF7 to SF12) to be received even if they use the same channel simultaneously. For thorough information on LoRa(WAN), refer to [9]. Current LoRa networks cannot support more than a few hundred IoT-devices connected to the same gateway [10, 11]. This is mainly due to the Medium Access Control (MAC) protocol, as the LoRaWAN MAC protocol, which we refer to as LoRaMAC, is Aloha-like, *i.e.*, when a device has a data-packet to transmit, it sends it immediately interfering any on-going communication [12]. It is well-established that the maximum achievable normalized throughput in a saturated network is 0.18 [13, 14]. For higher levels of traffic, the number of

International Conference on Embedded Wireless Systems and Networks (EWSN) 2020
17–19 February, Lyon, France © 2020 Copyright is held by the authors.
Permission is granted for indexing in the ACM Digital Library
ISBN: 978-0-9949886-4-5

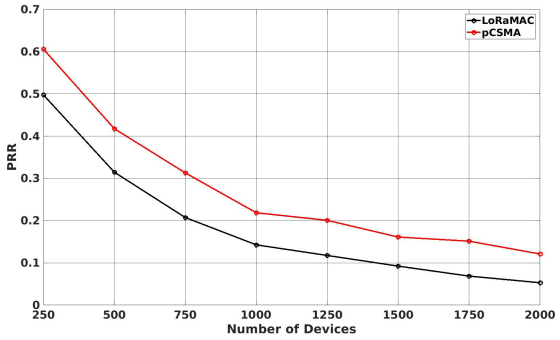


Figure 1: Packet Reception Ratio for increasing number of nodes for different MAC protocols - LoRaMAC (Aloha-like) and p-CSMA based with $p = \frac{1}{N}$. The devices transmit packets of 20 B periodically at 868.1 MHz using SF10.

Table 1: Comparison of Carrier Sensing approaches

	LoRaMAC	**p-CSMA**	**Static p-CARMA**	**adaptive p-CARMA**
Low consumption	✓	✗	✓✓	✓✓
Low overhead	✓✓	✗	✓	✓
Low complexity	✓✓	✓	✓	✓
High PRR (effectiveness)	✗	✓✓✓	✓	✓✓
High channel utilization	✗	✓✓✓	✓	✓✓

collisions increases rapidly, mandating a more-complex solution to at least double the number of devices per gateway.

Carrier-Sensing Multiple Access (CSMA) based schemes are widely used as they have been proven to decrease collisions and increase the channel goodput [15]. However, we cannot blindly apply existing CSMA techniques out-of-the-box on LoRa-devices due to the unique characteristics of LoRaWAN, which pose the following constraints:

- The long communication ranges between end-devices and a gateway result in the creation of several hidden sectors of devices [16].
- LoRa networks are usually unidirectional, without an immediate feedback channel from the gateway to each end-device, ruling out Acknowledge messages and Automatic Repeat reQuest (ARQ) mechanisms. The feedback from the gateway is far and few with one packet in several hours.
- LoRa-devices are energy constrained, and (repeated) carrier sensing consumes a lot of extra energy compared to Aloha-like schemes.

In this paper, we focus on increasing the number of IoT-devices served by a single gateway in a LoRaWAN from hundreds to thousands. To this end, we propose that devices should be *polite* and apply Carrier-Sensing (CS) on the MAC layer of LoRaWAN (i) to reduce the collisions, and (ii) to maximize the channel utilization in the network while being energy efficient. Although LoRa-devices do not have a sophisticated carrier-sensing mechanism, we can exploit their Channel Activity Detection (CAD) functionality, wherein packet preambles are detected, to check if the medium is idle. As packets can take a long time to transmit (several ms for larger spreading factors) CAD will introduce false negatives as no preamble will be detected during the transmission of the payload. To remedy this -phrasing it politely- imperfection, we combine CAD with the principles of persistent-CSMA (p-CSMA) [17] as it can be made to operate within the constraints listed above. We design a protocol, called p-persistent Channel Activity Recognition Multiple Access (p-CARMA) that tries to evade collisions with (most) neighboring devices using medium-sensing and probabilistically minimizes collisions due to true and fake (false negatives) hidden terminals using a localized and adaptive persistence p-value algorithm.

Fig. 1 shows the potential of being polite, yet persistent. It presents the simulated Packet Reception Ratio (PRR) of standard LoRaMAC and a naive CAD-based version of p-CSMA for an increasing number of devices (for details of our *ns*-3 simulation setup see Sec. 4). Without loss of generality, the value of p, determining the probability that a device transmits once the channel is found clear, is set to $1/N$ for large N devices. That probability would be optimal when all devices are within range of each other and carrier sensing is working perfectly, which is not the case in a star topology LoRaWAN and where CAD only looks for preambles instead of any activity on the channel. Nevertheless p-CSMA (red curve) increases the number of devices that can be effectively supported at a desired PRR with quite a margin compared to standard LoRaMAC (black curve). For example, for a PRR of 0.3 the number of devices scales from 500 to 750. Despite this 50% gain there is lot of room for improvement as the corresponding channel utilization is only 9.44%.

Adapting p-CSMA to LoRaWAN whilst guaranteeing maximal PRR is non-trivial due to the following challenges:

(a) p-CSMA is most effective when there are no hidden terminals, which is not the case in LoRaWAN. When using CAD, up to 44% of channel probes fail to detect an ongoing transmission (see Section 3.3).

(b) p-CSMA techniques use Acknowledgements to effectively establish a feedback control loop. These mechanisms, however, are not feasible in LoRa networks as they produce overhead in an already congested network.

(c) Contrary to p-CSMA, messages from the LoRa gateway are rare, not favoring centrally coordinated approaches to determine values of p.

(d) LoRa devices are blind to each other thus no coordination can be expected among them.

This paper aims to increase the scalability, specifically raise PRR when the number of devices increases, while being energy efficient, by reducing collisions using p-CSMA principles in a network with a large number of hidden terminals with minimum feedback. Our approach involves devices *indirectly* estimating the p value by sensing the channel using CAD, and learn about the collective channel occupancy. Based on this, each device independently selects a persistence value (p) deciding when to transmit in order to increase the chance of a successful transmission. Table 1 sums up the comparison of our adaptive p-CARMA protocol to classical (n)p-CSMA, naive CAD, and LoRaMAC, showing that p-CARMA is the preferable protocol to increase scalability under the constraints given by LoRa networks. Notice that vanilla LoRaMAC consumes more energy since the nodes simply transmits packet which requires more

energy than sensing (CAD) and dropping a packet.

Our contributions are the following:

1. We design p-CARMA, an adaptive, distributed p-value CS MAC protocol for LoRaWAN.
   (a) p-CARMA uses the CAD mechanism to increase the number of devices by at least twofold as compared to LoRaMAC.
   (b) In p-CARMA the gateway assists devices in adapting p-value by clustering its observations regarding the delay of packet deliveries.
   (c) This protocol can be used on existing LoRaWAN deployments without requiring any changes in the gateway/infrastructure.
2. We evaluate the performance and limitations of the CAD mechanism for channel sensing by conducting field experiments, and provide several findings therein.
3. We created several modules for simulations of LoRaWAN with CARMA capabilities in *ns*-3 in order to simulate several scenarios involving thousands of devices including energy measurements. The CAD module employs the results found in our real-world experiments for enabling a closer-to-reality simulations.
4. Through our extensive simulations, we provide key insights on p-CARMA's performance. We have also developed two metrics that must be used for evaluation of CARMA-based protocols in LoRaWAN.

The rest of the manuscript is organized as follows. A summary of the literature is provided in Sec. 2. In Sec. 3, first the on-field evaluation of the CAD mechanism is presented and then we describe the adaptive p-CARMA. In Sec. 4, the specifics of the simulated scenarios and the evaluation metrics are explained, and in Sec. 5, the results are presented along with discussion over the important observations. Finally, Sec. 6 concludes this manuscript.

2 Related Work

CSMA based techniques have been heavily investigated in the literature since it was first introduced by Kleinrock and Tobagi [17]. We first look at relevant works on CSMA and then discuss other MAC techniques attempted in LoRaWANs.

General CSMA Approaches. Kleinrock and Tobagi [17] introduced three variants of CSMA: (a) 1-persistent, (b) non-persistent and (c) p-persistent. A device continuously senses the medium until found idle and then transmits immediately in the 1-persistent case; in the non-persistent case, a device backs off for a random duration when the medium is sensed busy and transmits when the medium is sensed free; and in the p-persistent case, a device continuously senses the medium until it becomes idle and then transmits with probability p. Inherent assumption in these protocols was that devices are in the sensing range of each other. They relaxed this assumption in their subsequent work and tackled the hidden-terminal problems [16]. The authors proved the degradation of CSMA's performance, and found throughput-bounds for the 1-persistent CSMA. Let us now consider some important sophisticated CSMA variants, combined with other techniques, e.g., TDMA. A comparison of these variants based on five evaluation characteristics is in Table 2.

CSMA/SF favors transmission of packets with shorter payload, so that the corresponding transmitting devices do not sense the medium longer [18]. To evade the starvation of devices transmitting long packets, every device has internal clock. When this clock timeouts, the device segments its packet and retransmits the shorter payloads. However, keeping one extra clock per device and updating its timeout when other devices join/leave the network makes this protocol inflexible. In [19], CSMA is utilized to perform spectrum sharing in wireless multi-hop networks in a distributed way, where each device uses local information (direct communication with its neighbors), and local optima in terms of throughput are achieved. However, in LoRaWAN the communication follows a star-topology, where each LoRa-device has no information regarding its neighbors. In some works, each wireless device adapts its probability to access the medium as a function of the number of packets it keeps in its buffer [20, 21]. If the medium is sensed busy, devices do not transmit, but if sensed idle, the probability of transmission depends on the percentage of their buffer that is occupied. Generally, in LoRaWAN buffering more than one packet is not considered for uplink since every time a device keeps at most a single frame for transmission.

Table 2: Comparison of MAC protocols for LoRaWAN

	TDMA [26]	**Hybrid** [27]	**Multi-hop** [28]	**Adaptive p-CARMA** (this work)
Low consumption	✓	✓	✓	✓
Low complexity	✗	✗	✗	✓
Low cost	✓	✓	✗	✓
Low overhead	✗	✗	✗	✓
Distributed	✗	✗	✓	✓

Several works have considered combining CSMA with TDMA, or a type of time-slotting in order to enable Multiple-Access for applications such as, Machine-to-Machine (M2M) communications. Liu *et al.* proposed two phases for MAC: a contention phase, in which the devices try to claim the medium by performing p-CSMA, and a transmission phase, in which the successful devices of the previous phase transmit their packets by utilizing TDMA [22]. Shrestha *et al.* developed a channel access scheme that evaluates the throughput and energy consumption of devices in order to decide the preferable MAC technique, CSMA, TDMA, or a combination [23]. Shen *et al.* consider four groups with different delay-tolerance requirements, and they utilize Time Division Multiple Access (TDMA) for the two non-prioritized groups and p-CSMA with high p-values for the prioritized devices [24]. Several constraints in LoRaWAN limit the applicability of TDMA or hybrid approaches [25, 14]: (a) Class A of LoRaWAN is designed for uni-directional uplink traffic with only occasional downlink messages; (b) communication and synchronization overheads are a burden on the resource-constrained devices; and (c) time-division methods are known to be inflexible - when packet sizes are different and nodes join and/or leave, the slot assignment needs to be revisited and re-distributed to nodes in the worst case.

MAC protocols proposed for LoRaWAN. Hitherto, the related literature has been mostly focused on exploring the limits of LoRaWAN. Only a few works propose new MAC protocols for LoRa networks, mainly, in three categories: (i) TDMA, (ii) hybrid approaches, and (iii) approaches requiring extra modules (multi-hop LoRaWANs).

MAC on Time (MoT) [26] is a TDMA based protocol in which channel is divided into a reporting phase and a connection phase, with the latter being divided into time-slots, in which frames of fixed size are sent. However, LoRa-devices transmit packets of different sizes. Hybrid approaches combine TDMA and another protocol. Rizzi *et al.* [27] propose Time-Slotted Channel-Hopping for LoRa-devices *i.e.*, performing TDMA, and then frequency hopping per time-slot. While by using frequency-hopping and SF-orthogonality the authors allow more devices to access the medium, they do not improve the scalability of LoRaWAN, but rather test the limitations of TDMA under the current LoRa-capabilities. The multi-hop approaches involve clustering of devices, requiring

cluster-heads to forward packets to the gateways [28]. In clustering approaches the LoRa-devices need to be partitioned and cluster-heads with bi-directional connectivity to the devices and gateways need to be added. This increases the system complexity due to clustering, increases message overhead, is costly due to the need of extra modules with high energy needs (cluster-heads), and to a large extent does not augur well with the principle of LoRaWAN to be plug & play.

Pham proposed to adopt CSMA methodology used in IEEE 802.11 and IEEE 802.15.4 to improve the CAD in LoRaWAN [29]. He specifically, tries to map the DIFS and the back-off window duration to different number of CADs. However, an in-depth evaluation of fixing number of CADs has not been studied. We introduced the idea of applying persistent-CSMA in LoRaWAN to improve network scalability, and presented a preliminary implementation of p-CSMA in *ns-3* [30].

3 Design of *p*-CARMA

In this paper, we propose to adopt *p*-persistent CSMA for Channel Activity Recognition Multiple Access (called *p*-CARMA), as this approach is less aggressive to transmit packets than 1-persistent CSMA and more aggressive than non-persistent CSMA. *p*-CARMA cannot eliminate collisions due to hidden terminals, because solutions such as Request-To-Send/Clear-To-Send (RTS/CTS) that require two-way communication between a LoRa device and the gateway are not possible in LoRa. Despite this, *p*-CARMA can achieve a higher PRR than LoRaMAC as it attempts and often succeeds in choosing the 'right' moment to transmit. *p*-CARMA reduces the possibility of collisions with devices within its sensing range through CAD; and more collisions would be reduced by deferring transmissions due to the choice of '*p*'.

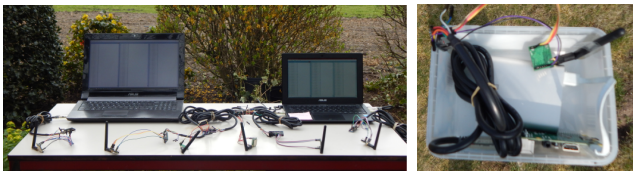
3.1 Design goals

While targeting better scalability than the current LoRaMAC, it is desirable that the proposed protocol follows the design principles of LoRaMAC. These are:

- **Distributed MAC**. Every device must be capable of making its decision on when to transmit without the help of a central controller/gateway.
- **Unslotted medium access**. Since there is no coordination possible from the gateway and each device can send packets of any size (within the limits), the devices must support unslotted medium access.
- **Simplicity**. The MAC must be of low complexity in order to be implemented on hardware with very low computational capabilities.

Any new MAC protocol that adheres to the above design goals will be inter-operable with the currently deployed LoRaWAN infrastructure. Additionally, we add the following design goals to increase the performance of LoRaWAN.

- **Increase scalability**. Scalability, here, refers to the ability to add or remove devices when the network is in operation, without affecting the operation and performance significantly. It may be argued that since Class-A LoRa devices hardly coordinate with the gateway for their operation, adding/removing devices can be done easily. However, the performance reduces with LoRaMAC when more nodes are added to the deployment due to collisions. In other words, we aim to maximize PRR through collision avoidance.
- **Minimize energy overheads**. Any newly introduced function must not consume significant amounts of energy as these devices are energy constrained.



(a) Receivers (b) Transmitter

Figure 2: The experimental setup with seven SX1276 devices interfaced with Arduino Pro Minis.

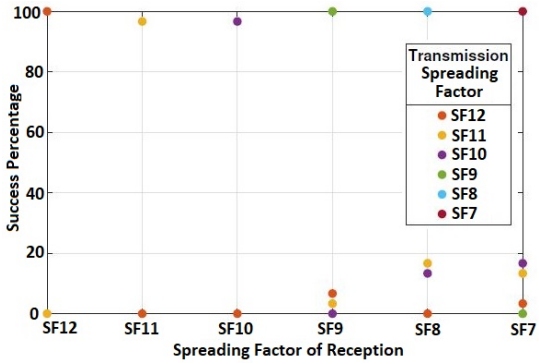


Figure 3: Success percentage of CAD when detecting preambles. The figure also shows the detection percentage across SFs; *e.g.*, when the receiver was listening on SF7 it could also detect SF10 with ≈19% success.

3.2 Basics of CAD

We use Carrier Activity Detection (CAD) for carrier sensing. Before proceeding to understand its limitations we briefly look at its workings. CAD was designed to detect the presence of LoRa preambles on the medium with the best possible power efficiency on LoRa devices such as SX1272/SX1276. The CAD mechanism was designed for choosing the non-interfering time to send the frame and for operating receivers in duty-cycle mode [31].

The CAD mode in LoRa consists of two phases: reception phase and signal processing phase. Once in CAD mode reception phase, a LoRa device switches its radio to receive mode on a pre-configured spreading factor (SF) and captures all symbols present in the channel. In the signal processing phase, the LoRa radio modem searches for correlation between the received waveform of symbols and the ideal waveform of preamble symbols. CAD detects only the preamble, whose typical length is 8 symbols. Clearly CAD has *not* been designed for carrier sensing as seen in the literature[17]. We, however, exploit this mechanism for *p*-CARMA in order to reduce collisions to the largest extent possible. Therefore, we need to characterize the performance of CAD before employing it in our design.

3.3 Characterization of CAD

3.3.1 Timing

Time required for the execution of a CAD loop depends on the SF and the bandwidth used. The exact relationship is found in [31].

3.3.2 Energy consumption

Our measurements with an SX1276 node reveal that around 11.5 mA is required for the radio reception phase and 6 mA for the signal processing phase for a bandwidth of 125 kHz. For a bandwidth of 250 kHz, these values increase to 12.4 mA and 6.8 mA for reception and processing, respectively. The total energy consumption depends on the SF used and can be calculated based on the

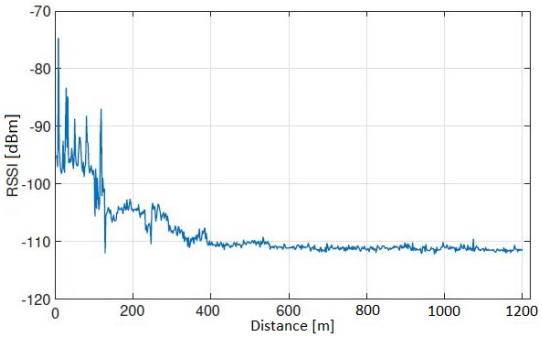


Figure 4: RSSI over distance from the receiver for SF9.

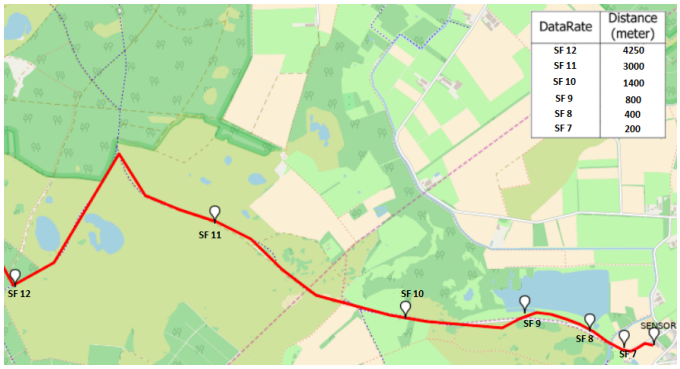


Figure 5: A map indicating the route taken (red line) and the sensing ranges for different SFs. The receivers were stationary (indicated by the point 'SENSOR' in the figure) and the transmitter was on a bicycle. The calculated ranges are the straight-line distances (not the distance the bicycle traversed).

time required, *e.g.*, for SF12 with a 125 kHz channel, a CAD cycle consumes around 1.8 mJ.

3.3.3 *CAD detection*

To understand the performance better, we characterize the accuracy and sensing ranges of CAD. We, therefore, setup a few experiments in a rural area free of any LoRa signals in the south of the Netherlands, where the terrain is flat as well. We used seven devices with SX1276 LoRa chipsets, with one device as transmitter and remaining six devices as receivers; each configured to receive on a certain SF. The receivers were placed at a height of 1.74 m and the transmitter was mounted on a bicycle 1.5 m above the ground. The advantage of this experiment is that it requires no sophisticated gateway since we are only evaluating the performance of an end-device for CAD. The experimental setup is presented in Fig. 2. Measurements were taken every 200 m. At each point, the mobile device transmitted 50 packets per SF.

Observation #1. The accuracy of preamble detection was at least 96% (see Fig. 3) when the receiver was configured to listen on the same SF as the transmitter. The 4% error creeps in mostly due to phase offsets.

Observation #2. For SF7, 8, and 9, there is a non-negligible percentage (as high as 19%) of detecting a preamble transmitted on a higher SF than the one that the receiver is configured to listen on (see Fig. 3). This is mostly due to the large preamble duration of SF10, 11, and 12, which can be captured by receivers of lower SFs.

Observation #3. While performing CAD on the payload part of the packets, the number of false negatives were found to be as high as 44%. This renders CAD unusable for detecting frames after preamble.

Observation #4. We found that the noise floor had an average RSSI value of -100 dBm. Fig. 4 shows the RSSI variation over distance for SF9. Clearly, RSSI cannot be used to reliably sense the channel as LoRa packets can be successfully received even when the signal power is below the noise floor [4].

Observation #5. Though the receiver was sensing during the preamble part of a transmitted packet, the CAD performance reduced as the distance increased between the transmitter and the receiver. The performance also decreased when more than one transmitter were active. We found that considering the results of three consecutive CADs compensated for such errors.

Observation #6. Fig. 5 shows the maximum distance for which a device could accurately detect the presence of a preamble on the channel. While SF7 preambles could not be sensed beyond 200 m, SF12 preambles were detected up to even 4250 m away.

3.4 *p*-CARMA

The characterization of CAD performance reveals that it is far from ideal to assess reliably if the medium is idle or busy. The problem is aggravated due to two facts: (a) false negatives creating lot more hidden terminals and (b) no unicast feedback per packet from the gateway, which eliminates adoption of a solution that solves hidden terminals such as RTS/CTS based mechanisms.

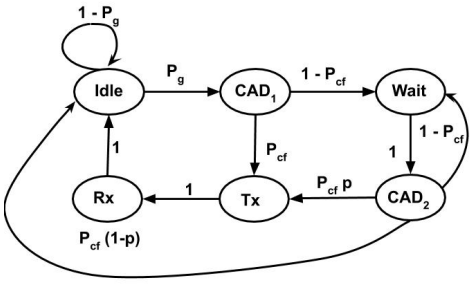


Figure 6: Markov model of *p*-CARMA protocol for a single device.

Reducing the probability of collisions is a pragmatic approach in this scenario. The two most promising candidates would be non-persistent and *p*-persistent CSMA techniques. Non-persistent CSMA can create large delays in transmitting messages, leading to lack of freshness of sensor data. Thus, we adopt *p*-CSMA, whose aggressiveness and collisions can be tuned by choosing the right value of '*p*'. Before we proceed to outline our distributed solution to set *p*, we describe the working of *p*-CARMA.

3.4.1 *Working of p-CARMA*

p-CARMA is a distributed, unslotted and low-complexity carrier-sensing based protocol. Every device performs the set of actions under specific probabilities, as denoted in the Markov model of the state transition diagram in Fig. 6.

A LoRa device begins its operation in the Idle state. When a new packet is generated and is ready for transmission with probability P_g, the device will perform CAD to assess the channel state (noted as CAD_1). If the channel is found free (with probability P_{cf}), the device will proceed to transmission (Tx). As we work with the class A devices of LoRaWAN, the node can open up two receive windows (Rx). Finally, it will return to the Idle state.

However, if the channel is occupied (i.e., probability $1 - P_{cf}$), the device proceeds to the state 'Wait'. In the Wait state, a random duration between 0 s and its payloads' average time-on-air (ToA) is picked, and then the device performs CAD (noted as CAD_2). As the device does not continuously perform CAD but waits (or sleeps) in

Algorithm 1: *p*-CARMA

```
   /* This function is called when a packet is
      generated                                          */
   Result: Transmit or drop the generated packet
1  Perform CAD_1
2  if channel found free then
3  |  Transmit the packet
4  else
   |  /* Sensed a preamble. Be polite to begin
   |     transmission after the current one.            */
5  |  pkt_tx_end_time = τ + ToA ;   // τ is the current
   |   time
6  |  backoff_time = min(τ + ToA * rand(), pkt_tx_end_time)
7  |  sleep (backoff_time)
8  |  Perform CAD_2
9  |  if channel found occupied then
   |  |  /* Sensed a new preamble within the
   |  |     on-going transmission.                      */
10 |  |  goto line 5
11 |  else
12 |  |  if τ < pkt_tx_end_time then
   |  |  |  /* No new transmission detected.           */
13 |  |  |  goto line 6
14 |  |  else
   |  |  |  /* No new transmission detected
   |  |  |     until the end of on-gonig
   |  |  |     transmission.                            */
15 |  |  |  Transmit the packet with persistence p
16 |  |  end
17 |  end
18 end
```

between, the energy consumption for CAD reduces.

As CAD only detects preambles, we would like to sense if there is any other transmission that begins within the on-going transmission, which is the reason for choosing back-off value between [0, ToA]. If the channel is still found to be occupied, an overlapping transmission is detected. The device gets back to waiting randomly before probing the channel. This repeats until the channel is found free. Upon finding the channel free, in order to be polite, the device will wait until the end of any on-going transmission. At the end of any such transmission, the device will either transmit (Tx) with probability p, or will refrain from transmission with probability $1-p$. In the figure, there is no packet queue/buffer considered. Therefore, the packet is dropped with probability $1-p$, and the device returns to the Idle state. This is shown in Algorithm 1. In case buffer is considered, the device returns to the Wait state with probability $1-p$.

3.4.2 Adaptive p-value Algorithm

When all devices are in the sensing range of each other it is easy to set p-values. In a network of N devices, the optimal value for p, assuming that the payload sizes are equal and have one packet to send, can be derived using Binomial distribution to $\frac{1}{N}$. This would, however, work optimally only when all the nodes are in the sensing ranges of each other; otherwise it leads to collisions and hence achieves only a sub-optimal result. Furthermore, as the p-value is not dependent only on 'N', when devices transmit under different periodicities sub-optimality is resulted.

However in LoRaWAN, not all devices are in sensing range or have same number of neighbors and CAD is not perfect. As the value of p is critical to the performance of p-CARMA, this value

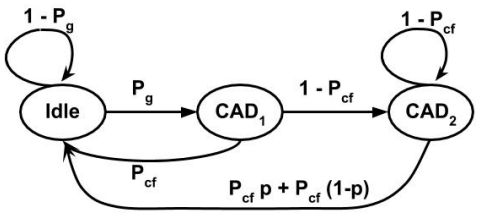


Figure 7: Simplified Markov model of p-CARMA protocol for a single device.

has to be chosen carefully.

The Markov model presented in Fig. 6 can be simplified to the chain shown in Fig. 7, where the transitions of probability 1 have been removed. This chain is evidently positive, recurrent and irreducible with transition matrix as follows:

$$P = \begin{array}{c} \\ \textbf{\textit{Idle}} \\ \textbf{\textit{CAD}}_1 \\ \textbf{\textit{CAD}}_2 \end{array} \begin{array}{c} \begin{array}{ccc} \textbf{\textit{Idle}} & \textbf{\textit{CAD}}_1 & \textbf{\textit{CAD}}_2 \end{array} \\ \begin{pmatrix} 1-P_g & P_g & 0 \\ P_{cf} & 0 & 1-P_{cf} \\ P_{cf} & 0 & 1-P_{cf} \end{pmatrix} \end{array}$$

Deriving the steady state distributtion will lead to the following equivalent equations:

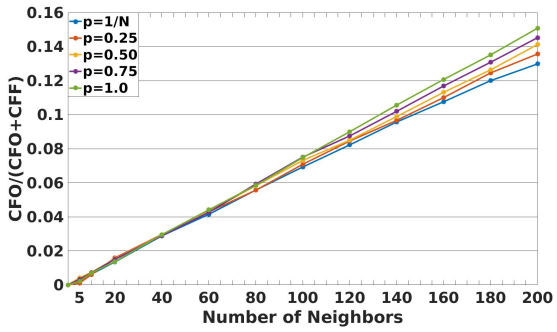
$$CAD_1 + CAD_2 = \frac{P_g}{P_g + P_{cf}} \quad or \quad Idle = \frac{P_{cf}}{P_g + P_{cf}} \tag{1}$$

The device being in sensing state, either CAD_1 or CAD_2, or in idle state is determined by unseemingly interdependent variables, *i.e.,* (i) the probability of generating a new packet P_g, (ii) the probability of finding the channel free P_{cf}, or occupied $(1-P_{cf})$ and (iii) persistence p. Usually, P_g is preset by users/operators. The number of devices (or traffic) in the network and the changes in the p-values of devices affect only P_{cf}. The p-values of devices can be viewed as knobs to be less or more 'aggressive' in transmissions, which affects the channel traffic and hence P_{cf}. This, according to Eq. (1), leads the device in changing the time it spends in 'sensing' state $(CAD_1 + CAD_2)$. In turn, any change on the time spent for CAD indicates that there are more or less devices transmitting (increased or decreased traffic) in one device's vicinity, leading it to update its p-value. By the above, it is obvious that we are led to a circular argument, as the network dynamics, i.e., transmission probabilities, hidden sectors of devices, dynamically changing traffic, do not allow an optimal solution regarding the p-values.

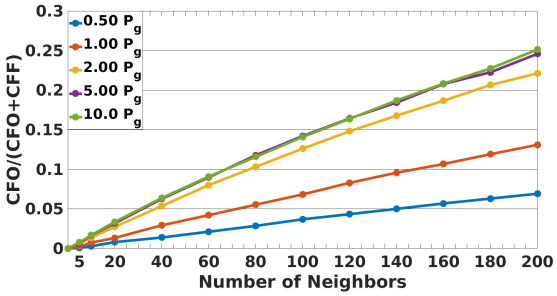
Furthermore, this model does not capture the dependency of other nodes in the network. The Markov model of Fig. 7 needs to be generalized into a model involving N devices. This would allow analytical procedure of deriving the p-value per device. However, such an analytical model cannot be derived due to the complexity as the number of possible states increases exponentially as $O(3^N)$ with the number of devices N. Thus, we are forced to determine the p-value adaptively by using a heuristic approach.

Heuristic Approach

To design a well-performing heuristic with the goal of improving PRR, we try to first understand the dependencies involved. From Eq. 1, the probability that each device senses the channel as free or occupied affects its sensing period, and as a result how the p-value is adapted. To adopt this observation in our heuristic, we focused on the number of times the device finds the channel free or occupied, CFF (Channel Found Free) and CFO (Channel Found Occupied), which are counted only at the first attempt of CAD for each packet, i.e., state transitions '$CAD_1 \rightarrow Tx$' and '$CAD_1 \rightarrow$ Wait' of Fig. 6.



(a) For increasing p-values



(b) For different packet generation probabilities and $p = 1/N$

Figure 8: Average ratio of number of times the channel sensed as occupied to number of devices in the range of each other, i.e., neighbors. SF10 and $P_g = 1.25\%$.

We evaluated $(CFO/(CFO+CFF))$ for increasing number of non-hidden devices in their vicinity (called 'neighbors'). The evaluation took place for p-values ranging from $1/N$ to 1.0 (cf. Fig. 8a) and for five different packet generation probabilities (cf. Fig. 8b).

Observation #1. As p-values increase, devices transmit more often, increasing the probability of finding the channel occupied and the term $(CFO/(CFO+CFF))$.

Observation #2. An increase in P_g leads to higher traffic and higher chance of finding the channel occupied. Specifically, in Fig. 8b increasing P_g by a factor of 2 corresponds to increasing the number of devices by a factor of 2 for the same P_g. However, for every p-value there is a threshold above which increasing the probability of generating packets does not affect $CFO/(CFO+CFF)$ even without adaptive p. For $p = 1/N$ this plateau is $P_g \simeq 6.5\%$. The main reason behind this counter-intuitive behavior is that the device senses only the preambles and not the payloads; the payload part is sensed as channel free of transmissions. This also explains for the low ratio seen in the figure, even when the channel is close to saturation (at least one packet in the air at any instant).

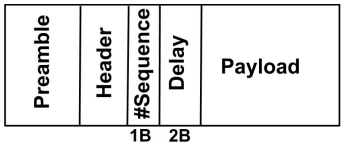
Observation #3. When the number of devices in the sensing range of each other (neighbors) increases, the ratio of sensing the channel as occupied increases almost linearly.

To apply the above observations in our heuristic, we use the complementary term $(CFF/(CFO+CFF))$, representing the probability to find the channel free on the first attempt based on observations. The equation of the adaptive p-CARMA is the following:

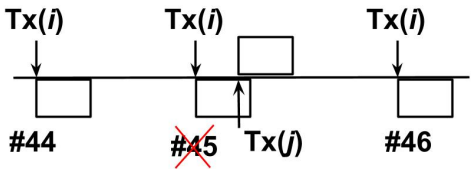
$$p = \Big(1 - CDR\Big)\Big(\frac{\overline{D} - D_{min}}{D_{max} - D_{min}}\Big)\Big(\frac{CFF}{CFO+CFF}\Big), \qquad (2)$$

with $1 \geq p \geq 1/N$.

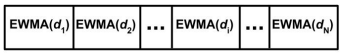
Since the term $(CFF/(CFO+CFF))$ is explained above, we proceed to explain CDR, $\overline{D}$, D_{min}, and D_{max}.



(a) Packet structure



(b) Sequence of events when a packet is lost due to collision



(c) EWMA of delays maintained at the gateway for N devices

Figure 9: Packet Structure and delay estimation of collided packets

Collision Delay Ratio (CDR) The CDR_i of a device i is the ratio between the total delay of its packets that were transmitted and collided, d_i^C, over the total delay of all three possible conditions for its packets[1]:

- Average delay between generation and transmission of successfully received packets, d_i^S (computed by the gateway).
- Estimated average delay seen by collided packets, d_i^C, (computed by the gateway).
- Average delay of the packets that were discarded due to persistence p, d_i^D, (computed by the devices).

In Fig. 9 (a), we show the simple packet structure, where we use a sequence number field and delay field, which help in finding CDR_i. To find the delay of successfully received packet d_i^S, gateway simply uses the 2 B delay field in the packet. Just before transmission, the device adds the time spent by the packet from its generation to the current time. It is a bit tricky to find the time (delay) the packet spent at the device before colliding at the gateway. Since we get no information from the collided packets, we resort to estimating the (possible) delays of the collided packets. For this, we keep a buffer for every device i that contains the exponential weighted moving average (EWMA) of the previously successfully received packet as shown in Fig. 9(c). Then using the sequence number, collided packets are found after receiving a packet successfully. Note, we assume transmitted packets are lost only due to collisions as LoRa is proven to be quite robust to channel conditions. Giving higher weightage to the delay experienced by the latest successful packet, delay of the collided packets are calculated by taking average of EWMA(d_i) and the d_i of the latest successful packet. Lastly, the devices can easily keep track of the delays experienced by the discarded packets. Consider the example in Fig. 9(b), where packet 45 is lost due to collision. The gateway computes EWMA after packet 46 is received with its delays. Now CDR

[1]We neglect packets lost due to error in reception but no collisions since such scenario is less likely.

is defined by,

$$CDR_i = \frac{\sum_{k=1}^{c} d_{i,k}^{C}}{\sum_{k=1}^{s} d_{i,k}^{S} + \sum_{k=1}^{r} d_{i,k}^{D} + \sum_{k=1}^{c} d_{i,k}^{C}}, \quad (3)$$

with c, s, and r being the numbers of collided, successfully received, and discarded packets. This could also be done just by counting number of packets collided, successful and discarded. However, using the delays can give a better picture of the dynamic nature of the events than just counting. Any increase in CDR_i denotes that less packets are received by the gateway from a device i. CDR is updated in stable periods, called 'observing period'. The observing period is the timespan between two ADR (adaptive data-rate) messages from the gateway to the device i. During this period, the gateway gathers the results of each device i regarding the total delays of its successful and its collided packets, d_i^S and d_i^C, and is decided by the users/system. In addition, each device is aware of the total delay for its discarded packets, d_i^D. At the end of each observing period, the gateway clusters the delay-data of devices, d_i^S and d_i^C, in three groups, (i) low , (ii) medium, and (iii) high delays [32]. The centroid C_{d^S} and C_{d^C} value of these three groups are communicated to each device. For example, if a device i belongs to low median group for successful case and high group for collided case, then gateway informs i the values of the centroids $C_{d^S}^L$ and $C_{d^C}^H$, respectively. All values are piggybacked on ADR messages in the downlink.

Now, since every device i knows its d_i^D, the individual CDR_i is computed and used in Eq. (2) throughout the next observing period in order for every device to adapt its p-value. During the next observing period, the gateway gathers again the delays (and in turn the centroids) to be used subsequently.

Parameters $\overline{D}$, D_{min}, and D_{max} These parameters are correspondingly the average, the minimum, and the maximum delay that packets experience in a device from the moment they are generated to the moment they are transmitted or discarded, i.e., states 'CAD_1', 'CAD_2', and 'Wait' of Fig. 6. Intuitively, $\overline{D}$ of a device is proportional to the number of devices (or traffic) in its sensing range. A value of $\overline{D}$ close to the D_{max} denotes that the traffic on the channel that the device can sense is high (given that many preambles are being sensed despite the considerable false negatives of CAD), i.e., the device spends most time between the states 'CAD_2' and 'Wait'. This implies that the p-value must be low. On the contrary, if $\overline{D}$ is close to the D_{min}, then the p-value must be high.

By bounding p to $1/N$, the device would at least have its 'fair' chance of medium access. Clearly, using Eq.2 can adapt to the traffic around the device. The total number of devices N, can be piggybacked by the gateway onto a downlink message. Note that to accommodate LoRa devices to join and leave the ambit of a gateway over time (usually days/months), we can recalculate the above parameters over a sliding window, which is a simple extension.

Network Bootstrapping During the network bootstrapping phase, each device starts with a pre-defined p-value, which can be in the range $[1/N, 1]$ (the value of p converges regardless of the starting value, which is shown in Sec. 5). The statistics for $\overline{D}$, D_{min}, and D_{max} are gathered over the first three transmissions. Then, all the parameters are updated every time a device manages to transmit a packet or refrains from transmission. Furthermore, from the bootstrapping and until the end of the first observing period, $CDR_i = 0$ for all devices. Thus, the devices still adapt based on local observations.

Table 3: LoRaWAN configuration used for simulations

Transmission Channel	868.1 MHz
Code Rate	4/5
Transmit Power	14 dBm
Bandwidth	125 kHz
Propagation loss model	Log-distance

4 Simulation Parameters and Metrics

Due to the scale of number of devices and also for the sake of repeatability for comparison, we had to undertake simulations for evaluating the performance of p-CARMA. Further, since EU guidelines strictly prohibit more than 1% duty cycle for LoRa devices, it is difficult to modify the behaviour of devices to send packets at higher rates to study our proposed changes. We considered the open-source code of [13] for *ns*-3 [33] to simulate according to the LoRAWAN specifications. To this code-base, we have developed CAD and extensible p-CARMA modules. Particularly, the CAD modules incorporate our findings from the field experiments of CAD (see Sec. 3.3). We shall make our modules open-source in due course.

We consider a scenario involving one gateway and all the LoRaWAN devices distributed uniformly around it. All the devices are using one transmission channel. To study the scalability, we simulate the above scenarios by increasing number of devices in steps. The details of the parameters and metrics used are given below.

4.1 Simulation parameters

We consider from 250 up to 3000 devices, with a step of 250, deployed around the gateway. Every device transmits periodically, with periodicity randomly chosen between the 1% duty-cycle limit and 3600 s. In each period, a device generates a payload of 20 B including 1 B sequence number and 2 B Delay (plus 8 B header).

The devices are assigned a spreading factor (SF) to transmit on, using Adaptive Data Rate (ADR) messages from the gateway [5]. Therefore, as in the real LoRaWAN, the farthest devices from the gateway use SF12 and the closest devices use SF7. The other LoRaWAN settings considered are listed in Table 3.

The energy values for CAD and devices are based on our measurements of SX1276 LoRa radios. The simulations were run for all the scenarios for 30 times, in order to get more than 95% confidence levels.

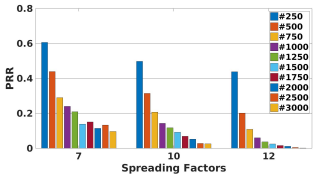
4.2 Metrics

We evaluated our simulations using the following metrics.

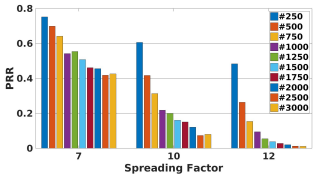
Packet Reception Ratio (PRR). The PRR is the ratio between the total number of packets successfully received at the gateway and the total number of packets transmitted. PRR evaluates the success of the MAC protocols despite collisions. MAC protocols are compared over the number of devices they serve for the same PRR-values, indicating how they scale. Furthermore, PRR indirectly evaluates energy efficiency as it denotes the ratio of received packets, for which the energy of transmission was not wasted.

Packet Transmittance Ratio (PTR). The PTR of a device is the ratio between packets transmitted and the number of packets generated by the device. The number of packets generated is equal to the number of packets transmitted if the device employs LoRaMAC as is. PTR is used to evaluate how much p-CARMA refrains from transmitting in order to reduce collisions.

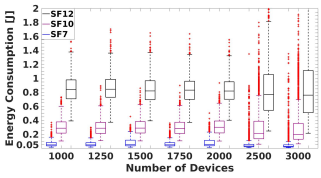
Received over Generated packets (RoG). RoG is the product of PRR and PTR, *i.e.*, it is the ratio between the total number of packets successfully received at the gateway and the total number of packets



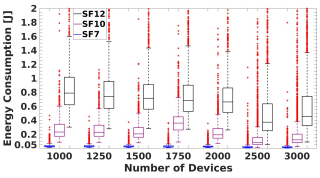
(a) LoRaMAC



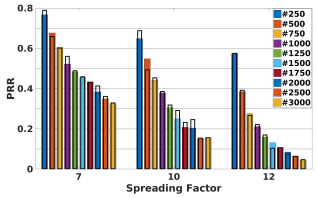
(b) static p-CARMA



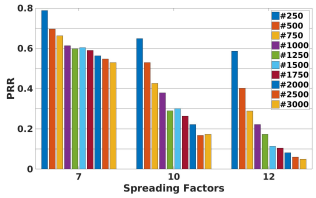
(a) LoRaMAC



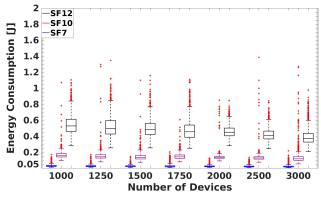
(b) static p-CARMA



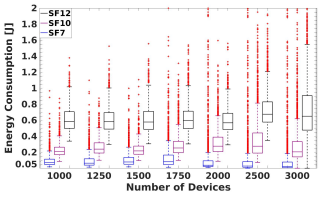
(c) adaptive p-CARMA, $p = 1, 1/N$



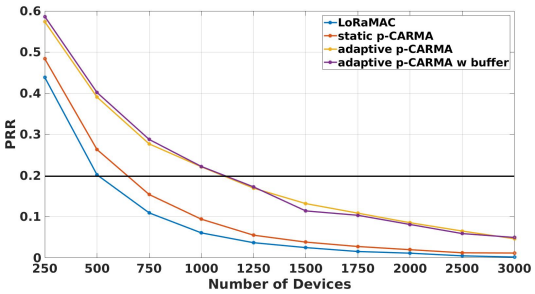
(d) adaptive p-CARMA with buffer



(c) adaptive p-CARMA



(d) adaptive p-CARMA with buffer



(e) performance for SF12

Figure 10: Average PRR for increasing number of devices.

Figure 11: Total energy spent for increasing number of devices.

generated. RoG presents the actual portion of the generated packets that reached the gateway.

Energy consumption. Energy consumption is monitored as an indication of the operational longevity of a device and to be compared with LoRaMAC. The total consumed energy of a device is the sum of the energy consumed for performing CAD operations, transmitting a packet and being in receiving mode.

Channel utilization. The time spent for packets that are transmitted and successfully received at the gateway is considered as an indicator of the efficiency in the usage of the channel.

5 Performance Evaluation

We evaluate the above metrics for the following protocols for comparison: (i) LoRaMAC (Aloha-like); (ii) p-CARMA with p fixed at $1/N$; we call it the 'static p-CARMA' approach that is nothing but the theoretically optimum value of p; (iii) adaptive p-CARMA without buffering and (iv) adaptive p-CARMA with devices using buffer of 1 packet. We consider only a buffer size of 1 since if a new packet is generated before the current one is transmitted, the older one is discarded. Further, for adaptive p-CARMA we consider an 'observation time' of 10 hours, i.e., the CDR of each device is updated per 10 hours. Furthermore, we show a subset of results for a few SFs for the sake of ease of representation and uncluttered presentation; the results for the other SFs also follow the same trend.

PRR. In Fig. 10, we compare the protocols with respect to PRR. First, we show the convergence of adaptive p-CARMA regardless of the starting p-value. The uncolored/transparent bars of Fig. 10c present the *PRR* of simulating adaptive p-CARMA under the same environment but for a starting p-value of $1/N$. It is apparent that the p-values converge and hence the PRRs converge as well.

p-CARMA (static and adaptive) clearly outperforms LoRaMAC in all the SFs and numbers of devices. p-CARMA improves the PRR of 3000 devices by a factor of 3.4 for SF7 and by a factor of 22.3 for SF12. Regarding the two p-CARMA approaches (static and adaptive), for SF7 they perform similarly. The static p-CARMA slightly outperforms the adaptive one by at most 1.29 times for 3000 devices. The adaptive p-CARMA outperforms the static for 250 and 1000 devices. This is due to the fact that SF7 devices are the ones closest to the gateway and are deployed as a circle. This creates a scenario with less hidden terminals. For the other SFs, the device deployment takes place in bands of rings around the gateway. In those cases, the hidden terminal issue becomes apparent, and the adaptive-p clearly dominates static-p for up to a factor of 5.25 for 2500 devices at SF12.

An important takeaway is that our adaptive p-CARMA performs at its best when the traffic/devices are higher, and hence the gains obtained for more than 2000 devices are seen to be the highest. This is because p-CARMA reduces collisions, even more than the static-p case. The adaptive p-CARMA with buffer has even higher PRRs, because the packets are not dropped with probability $1-p$, which creates higher traffic – the scenario better suited for adaptive p-CARMA. Using buffer outperforms static p-CARMA even for SF7.

A higher PRR does not necessarily mean higher scalability. However, for a given PRR, more devices can be accommodated by using p-CARMA. As indicated by Fig. 10e for SF12, adaptive p-CARMA achieves a PRR of 20% while accommodating double the number of devices compared to static approach and even more compared to LoRaMAC. Our metrics PTR and RoG that follow show that p-CARMA indeed transmits as many packets as possible while trying to avoid collisions whenever possible.

Energy. As we are interested in a scenario wherein there are numerous devices and many packets being transmitted, the subsequent graphs are for 1000 devices and more. In energy terms, the static-p strategy outperforms the LoRaMAC for high numbers of devices (above 1000). In particular, LoRaMAC consumes on an average 0.83 J per device regardless the number of devices for SF12, while the performance of static-p ranges between 0.40-0.79 J for the same case. While CAD is an overhead, as the number of deployed devices increases, CAD pays off due to reduced collisions and re-

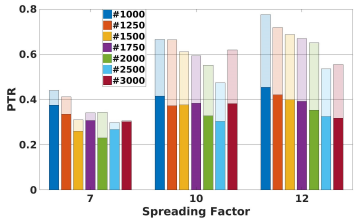
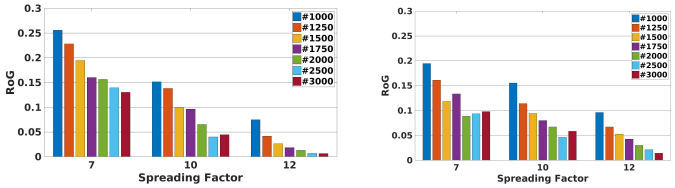


Figure 12: PTR (faded colours are for static-p)



(a) Static p-CARMA (b) Adaptive p-CARMA

Figure 13: RoG

duced transmissions. Consequently, the consumed energy is also reduced.

Adaptive p-CARMA outperforms both LoRaMAC and static p-CARMA in energy consumption, regardless of the number of devices. Specifically, the worst case of adaptive p-CARMA in SF7 consumes 0.4 times the energy of the corresponding LoRaMAC case. This ratio increases to 0.62 for SF12. The packet buffer case is not very different from the no buffer case for energy consumption, while the observed PRRs were higher. This implies that the adaptive p algorithm is effective and chooses the best time to transmit with its limited and imprecise information from CAD. While in back-off the devices sleep thus consuming negligible amount of energy.

PTR. The reduction in energy consumption that is achieved by adaptive p-CARMA is due to the transmission of fewer packets than the other two approaches. To evaluate the reduction in transmissions, we plot PTR for the static and adaptive p-CARMA. It is evident from Fig. 12 that adaptive p-CARMA in general transmits fewer packets than LoRaMAC (which is 1 in this figure). Furthermore, for high SFs the adaptive p-CARMA transmits considerably less packets than static p, catering to the increasing hidden-terminal issue. The effectiveness of our scheme is exemplified because of the higher PRRs.

RoG. The reduction in the number of packet transmissions is clearly visible when referring to the ratio between received and the total generated packets (RoG), as seen in Fig. 13a and 13b, respectively. Generally, for both p-CARMA approaches the ratio of received over generated packets varies from around 5% to around 17% depending on the number of devices and SF (see Fig. 13).

Although the average p-value is higher in the adaptive p-CARMA, the devices transmit fewer packets than with static p-CARMA that uses $p = 1/N$. This seemingly inconsistent behaviour is due to the fact that in p-CARMA, higher p-values means higher probability of non-hidden terminals transmitting and thus more devices refrain from transmitting. We outline an analytical explanation below.

Consider a 'sensing' device that has k devices in its sensing range. For the sake of simplicity, let us assume that the device can sense the entire packet reliably. These k devices performed CAD, but because another device in their vicinity, D, was transmitting, they found the channel occupied and are waiting for D to finish its transmission. D is not in the sensing range of our 'sensing' device. When D finishes, we will prove that the probability with which the 'sensing' device will sense a packet in the adaptive p-CARMA case, P_{ada}, is higher than in the static p-CARMA case, $P_{1/N}$. In the static p-CARMA, we have, $P_{1/N} = 1-(1-1/N)^k$, the probability of sensing a transmission. In the adaptive p-CARMA, we have, $P_{\text{ada}} = 1-\{(1-p_1)(1-p_2)(1-p_3)...(1-p_k)\}$. Since there are k elements in both P_{ada} and $P_{1/N}$, without loss of generality, $1-1/N \geq 1-p_1, 1-1/N \geq 1-p_2, ..., 1-1/N \geq 1-p_k$ (from Eq. (2)). Thus, we are lead to,

$$(1-1/N)^k \geq (1-p_1)(1-p_2)(1-p_3)...(1-p_k). \qquad (4)$$

and finally $P_{1/N} \leq P_{\text{ada}}$. This implies that there is a higher chance of sensing transmissions for the adaptive p-CARMA devices than for the static p-CARMA case. While the RoG values seem quite low, comparatively, adaptive p-CARMA achieves a significantly higher number of packets delivered successfully than LoRaMAC when the number of devices and/or SFs are high. This is observed in Table 4, where the successful receptions per 10,000 generated packets are shown. Even for cases of SF7 in which LoRaMAC has achieved more successful receptions, the number of failed transmissions is much higher compared to adaptive p-CARMA (cf. PRR, Fig. 10).

Table 4: Successful deliveries over 10,000 generated packets for 2000, 2500, and 3000 devices

(a) 2000 devices (b) 2500 devices

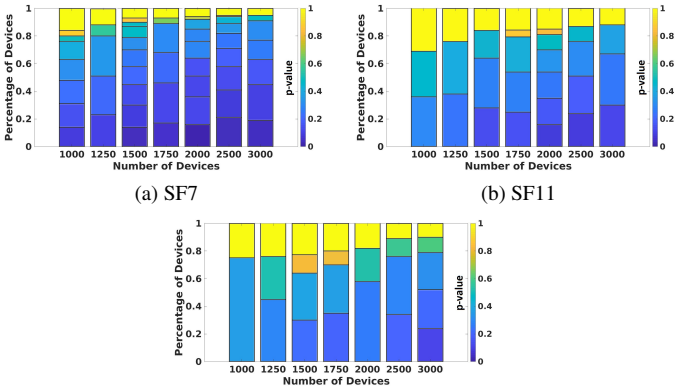
	LoRaMAC	p-CARMA	LoRaMAC	p-CARMA
SF7	**1148**	884	**1332**	935
SF10	529	**668**	280	**460**
SF12	114	**302**	50	**213**

(c) 3000 devices

	LoRaMAC	p-CARMA
SF7	969	**979**
SF10	260	**582**
SF12	20	**143**

Spread of p-values. Fig. 14 presents the spread of p-values across the number of devices and SFs. Each device starts from $p = 1$ and at the end of simulation, we grouped the resulting p-values applied around prominent p-centroids using the Expectation-Maximization algorithm (p-centroids are dictated by the color-bars). Using the centroids we will be able to observe the general trend of how p-values are converging. As seen, p-values tend to lower values when the number of devices in the network increases, due to the increased traffic which updates Eq. (2) more frequently. This indicates the adaptive operation of our algorithm to the devices (or traffic). For SF7, while 20% of the devices have p-values higher than 0.8 when the scenario involves 1000 devices, this number decreases to less than 5% when 3000 devices are involved. For SF10 and SF12, although the same trend is seen, the p-values are higher, due to the higher values of the term $CFF/(CFO+CFF)$ in Eq. (2), that is the result of the increased number of hidden terminals. Further, note that the standard deviation (SD) for all clusters with centroid $p = 1$ in Fig. 14 is 0, while for the other centroids $\text{SD}_p \in [0, 0.077]$.

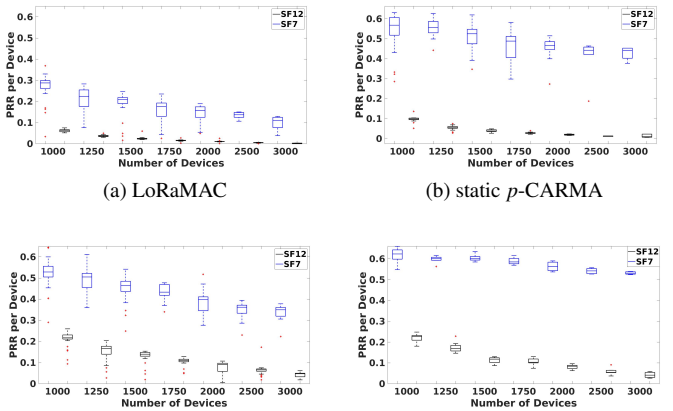
Variations in PRRs of each device. While there is a difference in



(a) SF7 (b) SF11

(c) SF12

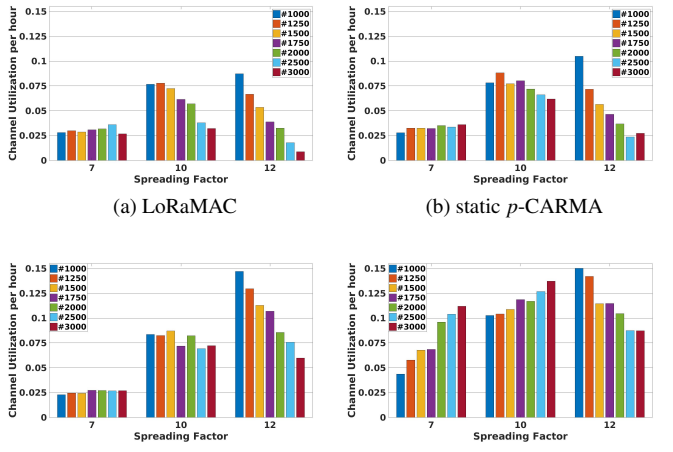
Figure 14: Spread of p-values versus number of devices



(a) LoRaMAC (b) static p-CARMA

(c) adaptive p-CARMA (d) adaptive p-CARMA with buffer

Figure 15: Average PRR per device for increasing number of devices



(a) LoRaMAC (b) static p-CARMA

(c) adaptive p-CARMA (d) adaptive p-CARMA with buffer

Figure 16: Normalized channel utilization for increasing number of devices.

p-values attained by different nodes because of different periodicity and their view of the channel, it is required to check how the PRR is achieved by each device. If the nodes use the channel politely, then they must have similar PRRs. It is important that the PRR across devices should not be heavily skewed even if certain devices have a higher p value. To ascertain this fact we use boxplots of PRR for various numbers of devices across all the devices in Fig. 15. LoRaMAC should inherently have almost similar PRR across all the devices since each device transmits as soon as it has a packet. In adaptive p-CARMA, the p values are chosen based on the perceived sensing activity. Therefore, the number of transmissions made per device is adapted. However Fig. 15 indicates that the PRR variations are extremely small in both static and adaptive p-CARMA, almost converging to a single p-value. Furthermore, as the collisions are reduced, the PRR achieved by each node converges. A significant aspect to note here is that we observe higher PRRs for higher SFs with adaptive p-CARMA. Further, the performance is increased even more when buffer is used, especially for SF7.

Channel Utilization. The efficiency of p-CARMA against LoRaMAC regarding the usage of the channel is observed in Fig. 16, especially for large numbers of devices and high SFs. For 3000 devices static p-CARMA outperforms classic LoRaMAC by a factor of 1.92 and 3.13 at SF10 and SF12 correspondingly. At the same cases, the adaptive p-CARMA is 2.24 and 6.86 times more efficient than LoRaMAC. Finally, when buffer is used, the effective channel utilization is not only increased even more for high SFs, but also for SF7, where we observe a considerable improvement for any number of devices.

6 Conclusions

Long Range Wide Area Networks (LoRaWAN) need to scale in order to cover large areas and to cater to large numbers of IoT-devices in the future, such as in Smart Cities. This work is focused on improving the scalability of LoRaWANs in an energy efficient way by making devices polite. To this point, we proposed a new MAC protocol of LoRaWAN, called p persistent Channel Activity Recognition Multiple Access, p-CARMA, which is based on the classical p-CSMA protocol. p-CARMA is distributed, less-complex, scalable and offers unslotted medium access to the LoRaWAN devices utilizing (imperfect) Carrier Activity Detection (CAD) for channel sensing. Real-world experiments were carried out to evaluate the performance and limitations of CAD in terms of distance and accuracy of preamble-detection.

Further, a complete *ns*-3 model was created to simulate p-CARMA on LoRaWAN. The design principle of LoRaMAC is to utilize ALOHA to ease the deployment and usage of IoT devices, however it is known to perform badly with more devices. Our adaptive p-CARMA was shown to achieve higher packet reception ratio (PRR) as well as higher number of received packets than LoRaMAC, especially when the number of devices/traffic is high, as p-CARMA reduces collisions. This is achieved through sensing the channel and choosing an appropriate value of p adaptively. For example, in a deployment involving three different orthogonal SFs (7, 10, and 12) and 2000 LoRa devices per SF having transmitted the same number of packets each, our adaptive p-CARMA manages to reduce packet collisions by 20% compared to the current LoRaWAN, consuming almost half the energy (0.48) on average per device.

Furthermore, it was shown that the energy consumption was lower than LoRaMAC even though additional CAD operations are executed. This is due to the number of transmissions, and consequently number of collisions, that are reduced to achieve better performance. Due to the adaptive nature of our proposed algorithm, PRR per device also converges.

The adaptive p-CARMA has been designed to perform better when the number of devices/traffic is high, and hence caters to handling scalability requirements. For low traffic scenarios and lower SFs, it is easy to see that static p-CARMA is sufficient with similar performance, however fails in real-world scenarios with large number of devices and increased traffic scenarios. The most important aspect of our adaptive p-CARMA is that it can be utilized on existing LoRaWAN deployments without requiring any changes in the LoRaWAN specifications but minor computation on gateways. The devices with p-CARMA can coexist with LoRaMAC devices making completely inter-operable. Finally, not only p-CARMA is applicable to classes-B/C, but it can offer improved performance (higher PRR) due to more assistance from the gateway.

Acknowledgement

This research was carried out within the SCOTT project (http://www.scott-project.eu) funded from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737422. This joint undertaking receives support from the European Union's Horizon 2020 research and innovation program and Austria, Spain, Finland, Ireland, Sweden, Germany, Poland, Portugal, Netherlands, Belgium, Norway.

7 References

[1] J. Bartje, "The top 10 IoT application areas based on real IoT projects," https://iot-analytics.com/top-10-iot-project-application-areas-q3-2016/, [Online; accessed: 2019-01-14].

[2] K. L. Lueth, "State of the IoT 2018: Number of IoT devices now at 7B - Market accelerating," iot-analytics.com/, [Online; accessed: 2019-02-22].

[3] M. Bor, J. Vidler, and U. Roedig, "LoRa for the Internet of Things," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*. Junction Publishing, 2016, pp. 361–366.

[4] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 60–67, October 2016.

[5] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent, "LoRaWAN^{TM} Specification," https://www.lora-alliance.org/portals/0/specs/LoRaWANSpecification1R0.pdf, Jan 2015.

[6] M. Bor and U. Roedig, "LoRa transmission parameter selection," in *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, June 2017, pp. 27–34.

[7] R. Sanchez-Iborra, J. Sanchez-Gomez, J. Ballesta-Viñas, M.-D. Cano, and A. Skarmeta, "Performance evaluation of LoRa considering scenario conditions," *Sensors*, vol. 18, no. 3, p. 772, Mar 2018.

[8] J. Petäjäjärvi, K. Mikhaylov, M. Pettissalo, J. Janhunen, and J. Iinatti, "Performance of a low-power wide-area network based on LoRa technology: Doppler robustness, scalability, and coverage," *International Journal of Distributed Sensor Networks*, vol. 13, no. 3, p. 1550147717699412, 2017.

[9] SEMTECH, "LoRa modem designer's guide," https://www.semtech.com/uploads/documents/LoraDesignGuide_STD.pdf/, [Online; accessed 11-Dec-2018].

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

[11] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa Low-Power Wide-Area Networks scale?" in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '16. New York, NY, USA: ACM, 2016, pp. 59–67.

[12] SEMTECH, "LoRaMAC," stackforce.github.io/LoRaMac-doc/, [Online; accessed 14-Nov-2019].

[13] D. Magrin, M. Centenaro, and L. Vangelista, "Performance evaluation of LoRa networks in a smart city scenario," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.

[14] K. Mikhaylov, J. Petäjäjärvi, and T. Haenninen, "Analysis of capacity and scalability of the LoRa low power wide area network technology," in *European Wireless 2016; 22th European Wireless Conference; Proceedings of*. VDE, 2016, pp. 1–6.

[15] R. Rom and M. Sidi, *Multiple access protocols: performance and analysis*. Springer Science & Business Media, 2012.

[16] F. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part ii - the hidden terminal problem in Carrier Sense Multiple-Access and the Busy-Tone solution," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1417–1433, December 1975.

[17] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part i - Carrier Sense Multiple-Access modes and their throughput-delay characteristics," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1400–1416, December 1975.

[18] G. Wang, K. Wu, and L. M. Ni, "CSMA/SF: Carrier Sense Multiple Access with Shortest First," *IEEE Transactions on Wireless Communications*, vol. 13, no. 3, pp. 1692–1702, March 2014.

[19] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 960–972, Jun. 2010.

[20] D. Shah and J. Shin, "Delay optimal queue-based CSMA," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, 2010, pp. 373–374.

[21] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length-based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 825–836, Jun. 2012.

[22] Y. Liu, C. Yuen, X. Cao, N. U. Hassan, and J. Chen, "Design of a scalable hybrid MAC protocol for heterogeneous M2M networks," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 99–111, Feb 2014.

[23] B. Shrestha, E. Hossain, and K. W. Choi, "Distributed and centralized hybrid CSMA/CA-TDMA schemes for single-hop wireless networks," *IEEE TWC*, vol. 13, no. 7, pp. 4050–4065, July 2014.

[24] W. Shen, T. Zhang, F. Barac, and M. Gidlund, "PriorityMAC: A priority-enhanced MAC protocol for critical traffic in industrial wireless sensor and actuator networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 824–835, Feb 2014.

[25] A. Pop, U. Raza, P. Kulkarni, and M. Sooriyabandara, "Does bidirectional traffic do more harm than good in LoRaWAN based LPWA Networks?" in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec 2017, pp. 1–6.

[26] G. Hassan and H. S. Hassanein, "MoT: A deterministic latency MAC protocol for mission-critical IoT applications," in *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*, June 2018, pp. 588–593.

[27] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, "Using LoRa for industrial wireless networks," in *2017 IEEE 13th International Workshop on Factory Communication Systems*, May 2017, pp. 1–4.

[28] F. A. Aoudia, M. Gautier, M. Magno, M. L. Gentil, O. Berder, and L. Benini, "Long-short range communication network leveraging LoRa and wake-up receiver," *Microprocessors and Microsystems*, vol. 56, pp. 184 – 192, 2018.

[29] C. Pham, "Investigating and experimenting CSMA channel access mechanisms for LoRa IoT networks," 04 2018, pp. 1–6.

[30] N. Kouvelas, V. Rao, and R. R. V. Prasad, "Employing p-CSMA on a LoRa network simulator," *CoRR*, vol. abs/1805.12263, 2018.

[31] SEMTECH, "Reading channel RSSI during a CAD," https://www.semtech.com/uploads/documents/an1200.21_std.pdf, [Online; accessed 29-Mar-2019].

[32] C. B. Do and S. Batzoglou, "What is the expectation maximization algorithm?" *Nature Biotechnology*, vol. 26, pp. 897–899, 2008.

[33] "ns-3," https://www.nsnam.org/, [Online; accessed 11-May-2019].