Channel Coding for Better QoS in LoRa Networks

Ulysse COUTAUD
Semtech Corporation and
LIG CNRS, Grenoble Alps University
Grenoble, France
ulysse.coutaud@univ-grenoble-alpes.fr

Bernard TOURANCHEAU
LIG CNRS, Grenoble Alps University
Grenoble, France
bernard.tourancheau@univ-grenoble-alpes.fr

Abstract—In the Internet of Things, Packet Delivery Ratio and Time on Air are two predominant characteristics for both applications and operators, especially while using transmissions over Low Power Wide Area Networks such as LoRa^{TM}. Our channel coding approach aims to improve these Quality of Service characteristics for LoRaWAN^{TM} networks. Our CCARR protocol uses Reed-Solomon FEC and structures successive frames into segments. A completion acknowledgements dynamically controls the amount of FEC overload. We estimate the potential gain of CCARR with a probabilistic analysis. Simulation and of-the-shelves testbed experiments of the protocol corroborate analysis trends and show a large Packet Delivery Rate improvement over LoRaWAN^{TM} and the literature with a controlled Time on Air increase due to optimized FEC overload.

Index Terms—IoT; LoRa^{TM}; LoRaWAN^{TM}; LPWAN; Channel coding; QoS; ToA; PDR; FEC; Reed Solomon codes.

I. Introduction

The rise of the Internet of Things (IoT) and its growing number of connected devices is a challenge for the next generation communication systems. Telecom and satellite machine-to-machine (M2M) technologies tried to answer the need, but cost, power and scalability concerns make actual solutions less appealing for the future [1]. The IoT applications are diverse and have diverging requirement: there is no wireless technology suitable for all scenarios [1], [2]. However, if IoT applications usually do not require strict delays, they often need Quality of Service (QoS) in terms of Packet Delivery Rate (PDR). For instance, in order to insure some accounting, especially in the metering domain. These smart metering systems are expected to connect a large number of end-devices, at a very low cost, in a wide area and with a decade battery-based autonomy. Their metering traffic consists in sporadic uplink transmission of short packets [3] and rare downlink maintenance transmission.

Low Power Wide Area Networks (LPWAN) offer a new solution to handle this challenge. But these networks present variable Packet Error Rate (PER) due to the wireless changing environment and their limited emission power. Various promising LPWAN technologies present different physical layers. Among them LoRaWAN^{TM} [1] developed by the LoRa^{TM} Alliance, is the subject of our study and experiments.

This paper presents the Channel Coding Adaptive Redundancy Rate protocol (CCARR). It attempts to improve the PDR of the state-of-art IoT LoRaWAN^{TM} networks by introducing Forward Error Correction (FEC) frames. This allows to reconstruct lost frames as soon as enough frames are received. A completion acknowledgment controls the overall transmission Time on Air (TOA) to avoid unnecessary redundancy.

The organization of this paper is as follows: Section II presents related works and necessary background. Section III give an overview of the LoRaWAN^{TM} technologies. Section V studies the LoRaWAN^{TM} Simple Repetition Rate protocol (SRR). Section VI presents our CCARR protocol that aims to improve PDR with channel coding in LoRaWAN^{TM} networks. Section VII and VIII present probabilistic and simulation studies of CCARR that allows analytical comparisons with SRR and literature works. Section IX describes CCARR implementation and performances obtained with an of-the-selves LoRaWAN^{TM} test-bed.

II. Related Works and Background

Forward Error Correction (FEC) consists in encoding an M symbols long dataword into a codeword using an Error Correction Code (ECC) which introduces N symbols of redundancy. The dataword can be recovered from the codeword as long as the number of erased symbols is less than e, one the ECC characteristics[1]. Reed-Solomon codes (RS) [4], are "perfect" ECC, meaning that their codewords are Hamming-distance equidistant, and are also systematic codes, ie. the dataword symbols are concatenated with the codeword symbols. RS-codes are able to correct $e = N$ erased symbols[2], but they are weak against burst errors. A method to handle this problem is to spread the codeword either over different channels [5] or in our case, over time with a buffered 2D data structure. A RS-code constructs a binary polynomial of degree M from dataword oversampling. Thus, from any large enough samples set, i.e. $\geq M$, the original polynomial can be deduced by interpolation technics. In practice, the decoding uses a less time consuming approach than interpolation (see [6]). In our work, we considered network frames as the

[1]ECC also corrects c corrupted symbols but it is not of our interest here since corrupted symbols are managed by layer two CRC in LoRaWAN^{TM}.

[2]RS-codes can also correct $c = \frac{N}{2}$ errors.

978-1-5386-6876-4/18/$31.00 ©2018 IEEE

symbols and we noticed that LoRaWAN^{TM} frames fits the erasure channel communication model assumed in RS coding because its physical layer handles a CRC. So for the upper software layer, a frame is either correctly receive or not received at all. Our CCARR protocol implementation uses an open-source RS-coding software described in [7]. However, as this program did not fit into micro-controller based platform, its adaptation is presented in Section IX.

Satellite communications and telecommunications use FEC technics for decades [6], [8]. LoRaWAN^{TM} SRR protocol implements frame emission repetition that we study in Section V. However, there are seldom works related to advanced FEC for LPWAN. The authors of [9] proposed a scheme, called DaRe, to improve PDR through FEC. DaRe computes redundancy over the application data with a convolution code adapted in order to reduce its complexity for end-devices. This channel coding achieves high PDR with rather low overload when the wireless channel parameters are well known. The convolution code approach uses the Channel State Information (CSI) from preceding downlink communications to tune its uplink FEC. However, we considered this a priori CSI knowledge to be a drawback in dynamic wireless transmissions such as LPWAN where CSI can vary from environment changes, network load and network server pushed parameters. Moreover, when CSI is not precisely known, DaRe's FEC must be set to an arbitrary high value implying very high overload even with low PER. On the contrary, our CCARR protocol does not need any precise channel parameters estimation. By design, it continuously adapts its FEC overload to match the decoding requirement with the channel evolution.

Several studies raise the question of the ability of LoRaWAN^{TM} networks to scale. Especially, the authors of [10] study the impact of uplink communications interference, where the performance of the system drops with the increase of end-devices i.e. the uplink traffic. Similarly, the authors of [11] study the negative impact of LoRaWAN^{TM} "confirmed" uplink frames and more generally downlink communication impact, on the LoRaWAN^{TM} network capacity. Our CCARR protocol introduces PDR QoS with less uplink FEC traffic than LoRaWAN^{TM} repetitions or DaRe but it needs a few downlink acknowledgements.

III. LoRaWAN^{TM} Long Range Communication

LoRaWAN^{TM} defines a communication protocol and system architecture based on the LoRa^{TM} physical layer which enables long-range and low power communication. As shown in Fig. 1, LoRaWAN^{TM} lays out a star-of-stars topology in which few always-on gateways relay secured messages between numerous battery powered end-devices and a back-end central network server. Notice that end-devices are not associated with a specific gateway. Thus a frame emitted by an end-device is typically received by multiple gateways which forward them to the server in charge of duplicate elimination, communication protocol decision, application delivery, etc.

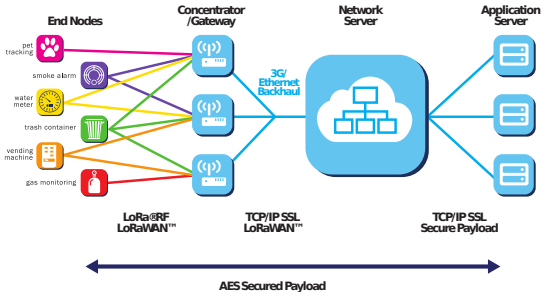

Fig. 1. LoRaWAN networking architecture from [12].

LoRaWAN^{TM} defines different device classes based on active listening behavior which drives the energy consumption, as detailed in [13]. Our work targets Class-C gateways with a nearly continuously open receive window, only closed when transmitting downlink and numerous, often energy autonomous, Class-A end-devices whose uplink transmission can be followed by two short downlink receive windows for optional acknowledgement and piggybacked control.

LoRaWAN^{TM} MAC channel access is a simple contention based ALOHA [14]. Its frame's structure contains synchronization preamble, header, payload and CRC. Apart from operator bands, in Europe, LoRaWAN^{TM} operates mostly in the 863-870MHz and 433-434.8MHz ISM bands, representing 70 channels of 125kHz bandwidth each. Most of these ISM bands regulation states that end-devices and gateways have a maximum 1% duty cycle in emission when using an Aloha-based access method. Notice that, as studied in [15], this is a significant limitation on the downlink channel that a gateway shares between all its end-devices neighbors.

LoRa^{TM} physical layer is a proprietary Chirp Spread Spectrum modulation (CSS) which trades data rate for sensitivity within a fixed channel bandwidth. LoRa^{TM} implements six data rates, using orthogonal Spreading Factors (SF), from 7 to 12. As these SF do not interfere with each others, they create six sub-channels per channel, increasing the capacity of the network [16]. As a consequence, with SF and emission power (TP), a LoRaWAN^{TM} system compromises between data rate, frame Time On Air (TOA), communication range, power consumption, interference level, and network capacity performances, within a constant bandwidth, see details in [13]. In order to illustrate this flexibility we analytically computed these trade-offs between transmission distance and bit TOA as shown in Fig. 2 using the Friis Free Space model in typical[3] conditions and 16 bytes long frame

[3]Noise floor calculated according to [16] at 19.85°C; Absolute receive power computed from the Friis Free Space model [17]; Path loss coefficient 2.64; Receiver sensitivity from [18]; Absolute transmitted power 14dBm; Frequency 868MHz with a 125kHz bandwidth.

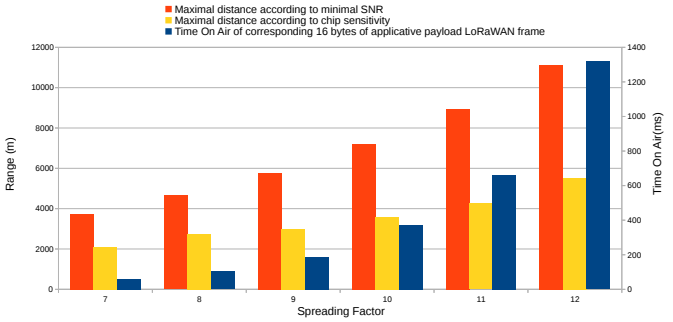

Fig. 2. LoRa^{TM} analytic tradeoffs between TOA and transmission range against spreading factors.

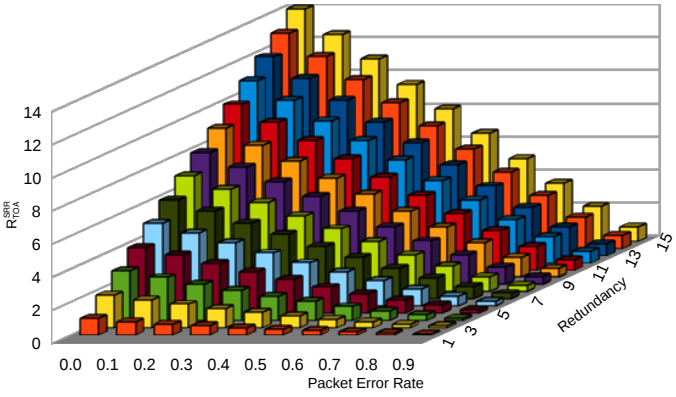

Fig. 3. R_{TOA}^{SRR} as a function of the PER and the repetition number in an ideal LoRaWAN^{TM} transmission protocol over an i.i.d. channel.

payloads.

The failure of a transmission may be caused by many factors : distance, path loss, noise, frame preamble length, frame length, etc., as well as frame collision without capture effect. This asserts the assumption that frames losses are independent and identically distributed loss (i.i.d) events. In the following, from our knowledge and experiments from [19] we consider that a good LoRaWAN^{TM} channel model is an independent and identically distributed loss (i.i.d) frame erasure channel with the same PER for uplink and downlink communications. The frame error detection is handled by the LoRa^{TM} physical layer CRC mechanism that insures the erasure of corrupted frames. Notice that [9] considers burst erasures, a case covered for free by our CCARR proposal based on segment of successive frames. Notice that downlink acknowledgements may experience better PER due to their shorter frame size and the greater gateway emitting power allowing capture effect.

IV. TOA comparison metric

The LPWAN capacity is an important issue for IoT networks operators. Thus reducing the overall transmission TOA is a priority, especially because our PDR QoS improvement introduces FEC. Let's define R_{TOA}, a comparison metric, that represents the channel TOA overload ratio, ie. the redundant transmissions duration that occurs while the data have already been recovered, over the transmissions duration that was necessary to complete these data reception:

$$R_{TOA} = \frac{TOA_{useless}}{TOA_{useful}}$$

Thus, the best channel usage results in $R_{TOA} = 0$, which means that all the transmissions that happened, including redundancy, were necessary to recover the data. Consequently, $R_{TOA} > 0$ reflects the wasted network capacity normalized relatively to the useful transmissions necessary in order to successfully transmit the data with the given protocol. For instance $R_{TOA} = 100\%$ arises when the data is re-transmitted while it was already recovered on the receiver.

V. LoRaWAN^{TM} Simple Repetition Rate

In order to improve PDR QoS, LoRaWAN^{TM} proposes to set a parameter $r \in [1..15]$, that determines the repetition number of frame emission (as described in [17]) called SRRr protocol. Hence, if we consider a channel[4] with independent and identically distributed frame loss (i.i.d.) and $PER = p$, the useful $P(TOA_{useful})$ value is obtained when all the preceding frame transmission attempts were not successful, thus it initially at least equals one, then p, p^2, etc. From the preceding remark and section IV, we then compute:

$$R_{TOA}^{SRRr} = \frac{(r - P(TOA_{useful})) \times TOA_{frame}}{P(TOA_{useful}) \times TOA_{frame}}$$

$$R_{TOA}^{SRRr} = \frac{r - \sum_{i=0}^{r-1} (p)^i}{\sum_{i=0}^{r-1} (p)^i} \quad (1)$$

Fig. 3 shows the evolution of R_{TOA}^{SRR} with the PER and repetition number acting together on the transmission efficiency. This shows that SRR introduces a lot of unnecessary network overload, when the channel's PER is not very high. However, R_{TOA}^{SRR} stays low when the PER is very high. Thus, this protocol is only efficient for very high PER.

Notice that a complementary protocol is proposed in LoRaWAN^{TM}, the Adaptive Data Rate (ADR). In ADR, the network server piggybacks downlink command bits which may adapt the end-devices transmission parameters in order to maximize end-devices battery life and network capacity usage. This protocol is not part of the scope of this study but rather a method that is supposed to be triggered in order to minimize potential interference by reducing TP and spreading. This decreases the link quality and thus fits within the SRR best efficiency zone.

[4]We assume in the following that the channel PER is the realization of the error probability distribution.

VI. Channel Coding Adaptive Redundancy Rate

However, in order to improve the PDR QoS, it is preferred to send just as much redundancy as required. A solution is to acknowledge every successfully received frame, using the Automatic Repeat reQuest (ARQ) error-control protocol. This corresponds to sending all frames as LoRaWAN^{TM} confirmed data. Unfortunately, such a solution causes several problems. In the best case (low PER) the uplink gain is mitigated by the downlink overload created by the acknowledgements. In the worst case (high PER), this solution may even be worst than SRR because when the uplink data are received by the server and the downlink acknowledgement is lost, eventually, both the end-device and the gateway will emit again, wasting network capacity and increasing interference. Moreover, in practise, this solution does not scale with LoRaWAN^{TM} ALOHA access in ISM bands, since gateways must respect the 1% duty cycle regulation no matter the number of end-devices.

With the aim of improving PDR QoS while maintaining the TOA of the wireless channel as low as possible, we propose in the following the Channel Coding Adaptive Redundancy Rate (CCARR). CCARR is a protocol that uses FEC at the frame scale, as in SRR and DaRe, and a completion acknowledgement for each fixed number of frames segment.

Obviously, CCARR reduces the downlink overload compared to ARQ, with only segment completion acknowledgements. Comparatively to DaRe, CCARR does not need any precise channel CSI estimation. By design, it continuously adapts its FEC overload until it reaches the decoding requirement, following the channel evolution. Hence, CCARR has a TOA overload inversely proportional to the channel's PER and leads to a better TOA than SRR for the same PDR. However, the price to pay in CCARR is segment based frequency downlink acknowledgements and the frame scale FEC coding on emitters as well as segment scale decoding on receivers. Such a protocol, as in DaRe, may increase application communication delivery delay but one can notice that small delays are not an issue in most of the IoT applications, such as metering, where low PDR is a major concern.

More precisely, in CCARR(n,m), the information to be sent is gathered in a $(n+m)$ frames long segment where each frame has l Bytes length. The first n frames contain application data and the following m frames contain redundancy data. Each redundancy frame is an RS-code computed vertically over the preceding frames as shown in Fig. 4. Notice that n is adjustable, for instance to the application requirement, the physical payload needs and also to adjust downlink channel load.

On the end-device, as described in Algorithm 1, the CCARR(n,m) protocol sends segment frames one by one.

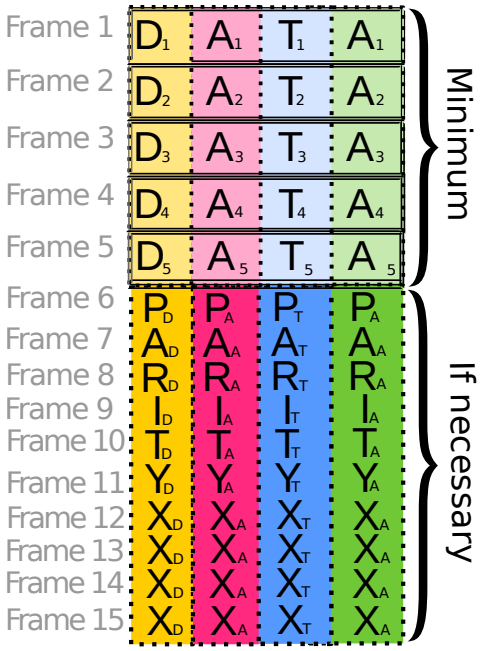

Fig. 4. Layout of a CCARR($n = 5$;$m = 10$) segment with $l = 4$.

```
f←0; ack←FALSE;
while (f < n + m) AND (!ack) do
    SendFrame(frame#);
    f++;
    if (f ≥ n) then
        ack←ListenForAck();
    end
end
if (f = n + m) AND (!ack) then
    LostSegmentError(frame# DIV (n+m))
end
```

Algorithm 1: CCARR(n,m) segment uplink emission on end-device.

It only starts listening for downlink acknowledgement[5] at the n^{th} frame transmission. If no such downlink acknowledgement is received, it then continues sending uplink redundancy data frames with active listening on acknowledgement windows until the reception of the downlink acknowledgement completion segment.

Thanks to RS-code properties, the network server will be able to retrieve the whole application data from any n received frames. As described in algorithm 2, the CCARR(n,m) network server only triggers the gateway downlink acknowledgement(s) emission to the end-device when it received n frames, i.e. enough to reconstruct the segment. Notice that the server keeps acknowledging when it receives new uplink frames for an already retrieved segment because this means that the previous

[5]In the two acknowledgement time slots defined by LoRaWAN^{TM} specification [17].

```
f ← 0; ReceiveFrame(frame#);
while (frame# DIV (n+m)=CurrentSegment#) do
    f++;
    if (f ≥ n) then
        SendAck();
    end
    ReceiveFrame(frame#);
end
if (f < n) then
    LostSegmentError(frame# DIV (n+m))
end
```

Algorithm 2: CCARR(n,m) segment reception treatment on the network server.

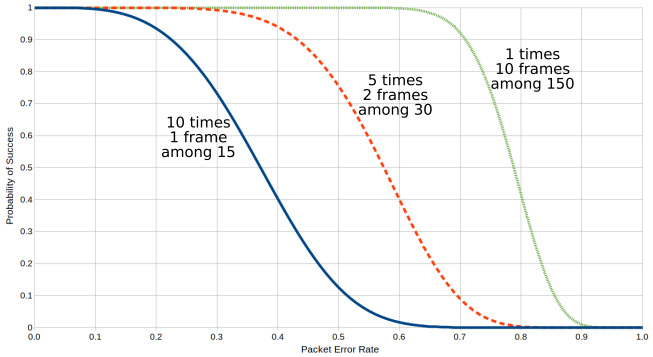

Fig. 5. Probability of reception for ten frames with different strategies requiring 150 attempts, against channel PER.

acknowledgement(s) was(were) lost. In the best case, CCARR(n,m) transmits only n uplink data frames and one downlink acknowledgement. If some frames are not received, CCARR(n,m) transmits necessary parity frames until the successful reception of n frames. Thus, the acknowledgement "covers" at least n frames and more than n successful data frame transmissions happen only when an acknowledgement is lost. If everything goes well, compared to the ARQ protocol, the number of acknowledgements is divided by n and compared to SRR, the number of data frames is divided by r, while compared to DaRe the FEC overload is dynamically adjusted to the channel evolution.

VII. CCARR Probabilistic Analysis

One can notice that the segment completion acknowledgement is triggered by any n frames receptions over $(n+m)$ sent. Thus, in the analysis with i.i.d. channel frame loss, this results in a cumulative binomial law for the PDR probability of success. This means that CCARR has a better probabilistic chance of successfully deliver the application data than SRR, because for the same amount of uplink redundancy CCARR's PDR is probabilistically better than the SRR's.

For k application data frames and a channel PER of $p = (1-q)$, we can write more formally:

$$\mathcal{B}_c(k,n,q) = \sum_{i=0}^{k} \binom{n}{i} (q)^i (1-q)^{n-i}$$

be the cumulative value of the binomial law $\mathcal{B}(k,q)$ when k varies from 0 to n, and q is the probability of success. Then, from [20], it is possible to deduce that :

$$\mathcal{B}_c(k, k \times z, (1-p)) \neq \sum_{i=1}^{k} \mathcal{B}_c(1, z, (1-p)) \qquad (2)$$

Moreover, from the computed values of Fig. 5, one can notice that with $k \geq 2$ and $p \leq 0.914$, inequality (2) becomes a strictly greater equation. This means for instance that the reception of ten frames over 150 attempts has a larger success probability than the reception of one frame over fifteen attempts repeated ten times.

Fig. 5 illustrates this phenomena for typical values which show the CCARR analytical advantage over SRR in the simplified context of this analysis. In most cases, for the same amount of redundancy, the CCARR protocol has a better probability to retrieve the whole data over a given channel than the corresponding sequence of SRR protocols. In order to quantify this gain, let's determine R_{TOA}^{CCARR}, the overload ratio as defined in Equation 1 and consider the case of i.i.d. frame loss with the same probability for uplink and downlink transmissions.

The cost of sending an uplink frame with CCARR depends on a_i, the TOA spent by the i^{th} frame transmission attempt, the probability b_i to reach this i^{th} attempt and the probability c_i that not enough successes occurred (i.e. less than n frames) to retrieve the whole data at this i^{th} attempt :

$$R_{TOA}^{CCARR} = \frac{\sum_{i=1}^{n+m}(a_i \times b_i \times c_i)}{\sum_{i=1}^{n+m}(a_i \times b_i \times (1-c_i))} \qquad (3)$$

Let T and T' be respectively the transmission time of a data frame and of an acknowledgement, then:

$$\begin{aligned} a_i &= T + B_c(n,i,1-p) \times (1-p) \times p \times T' \\ b_i &= \prod_{k=1}^{i-1} (1 - B_c(n,k,1-p) \times (1-p)^2) \\ c_i &= B_c(n,i,1-p) \end{aligned}$$

Fig. 6 shows the computed R_{TOA}^{CCARR} overload ratio to send uplink frames with CCARR, T and T' being typical LoRaWAN^{TM} values[6], computed against the channel's PER and the maximum amount of redundancy chosen. R_{TOA}^{CCARR} increases with the channel PER until a maximum, depending on the redundancy rate. This overload happens when the CCARR server does not succeed to acknowledge the segment soon enough, ie. the

[6]Notice that the maximum SF12 payload is 55 Bytes. In order to fit with our experimental constraints of Section IX, we present here the results for 29 Bytes frame and 16 Bytes acknowledgement. However staying in this range, we did no see noticeable differences.

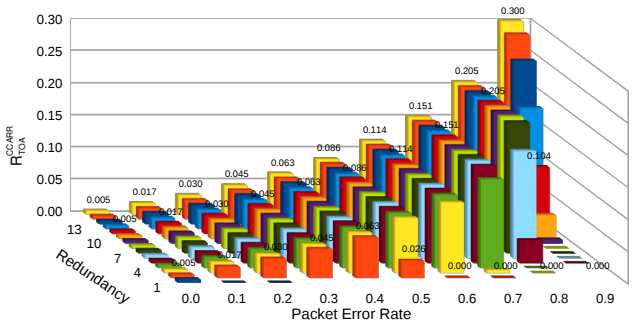

Fig. 6. R_{TOA}^{CCARR} in frame TOA for CCARR(100,100×(r-1)) against redundancy and PER.

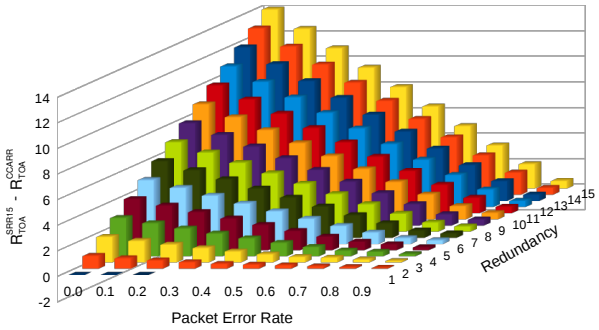

Fig. 7. (R_{TOA}^{SRR} - R_{TOA}^{CCARR}) in frame TOA for CCARR(100,100×(r-1)), against redundancy rate and PER.

acknowledgement is lost. Then the cost decreases, showing that more of the redundancy is useful to recover the data. Notice that the redundancy parameter varies up to 15, the LoRaWAN^{TM} SRR15 protocol maximum overload.

Fig. 7 explicitly displays the difference between the R_{TOA} ratios for SRRr and CCARR(10,10×(r-1), which present a comparable amount of redundancy, i.e. in the worst CCARR case, the exact same number of frames are sent. The values correspond to TOA improvements i.e. network capacity gain. Clearly, in this analysis context, CCARR sends much less frames for the same overall successful data transmission[7]. The gain increases with the redundancy because CCARR emits only the required amount of redundancy. The gain decreases with the channel PER, because a large amount of redundancy has to be sent in any case to recover the data.

VIII. Simulation Results

In order to see the limits of the CCARR analytic trends, we implemented a simulator in C to observe its behavior in various situations. In the following, all the simulation experiments represent the transmission of at least 400000 frames in order to get a 95% confidence interval.

[7]The very small negative values correspond to CCARR acknowledgement penalty when exactly the same number of frames occurs for very low PER.

The simulator runs three unix processes working together to emulate the end-device, the channel and the server. The end-device process reads data from a file, computes the segment and emits it using the CCARR protocol. The channel process drops or transmits the frames from the end-device process to the server process following the propagation model implemented. The channel follows the i.i.d. frame erasure channel model discussed in Section III. Its PER is configurable. The server process receives the frames and recovers the data through RS-decoding if necessary. All the simulator processes write their metadata in a log-file to allow post-mortem analysis.

For the sake of simplicity, we assume a fixed quality of the physical channel, i.e. the PER does not depend on time, nor frame's parameters (length, preamble length), nor uplink/downlink. Similarly, the simulator implements only one receive window after each transmission where the device listens over the emission channel. For coherency with Section IX constrains, the payloads are set to 29 Bytes frames and 15 Bytes acknowledgements.

CCARR is evaluated against SRR with its best robustness, SRR15, the maximum PDR QoS allowed by LoRaWAN^{TM} and is eventually compared with DaRe. The three protocols CCARR(a,a×(15-1), with $a = 1, 10, 100$ have a maximum redundancy amount equals to SRR15. The default practical value, SSR5, is represented as a reference in order to visualize the larger FEC gain. The simulation results of Fig. 8 compare the PDR reached by CCARR with the different segment sizes and the PDR obtained with LoRaWAN^{TM} SRR15 protocol against the channel PER. For CCARR(1,14), which is equivalent to the ARQ protocol, the PDR is logically similar to SRR15. As predicted by Section VII, CCARR PDR performance increases with the segment sizes. We distinguish three trends from the curves :

- Until 74% PER all the tested strategies but SSR5 offer enough redundancy to recover the data with a PDR of more than 99%.
- From 74% to 92% PER, CCARR gives a progressively better robustness than SRR15, and around 90% PER, CCARR(100,1400) obtains a 26% better PDR than SRR15.
- After 92% PER, CCARR(10,140) then CCARR (100,1400) PDR drops, and from there, the whole data is not recovered anymore, so the PDR corresponds to (1-PER).

Fig. 9 shows the TOA of CCARR and LoRaWAN^{TM} SRR protocols against the channel PER. For the sake of clarity, the results are expressed in uplink frame TOA. As expected, SRR15 and SRR5 show a constant TOA that only depends on their number of transmission repetitions. On average, the CCARR protocol largely reduces the necessary TOA as it stops transmitting as soon as the data is successfully received. For instance, CCARR's TOA with segment(10,140) is half of SRR15's over a 80% lossy

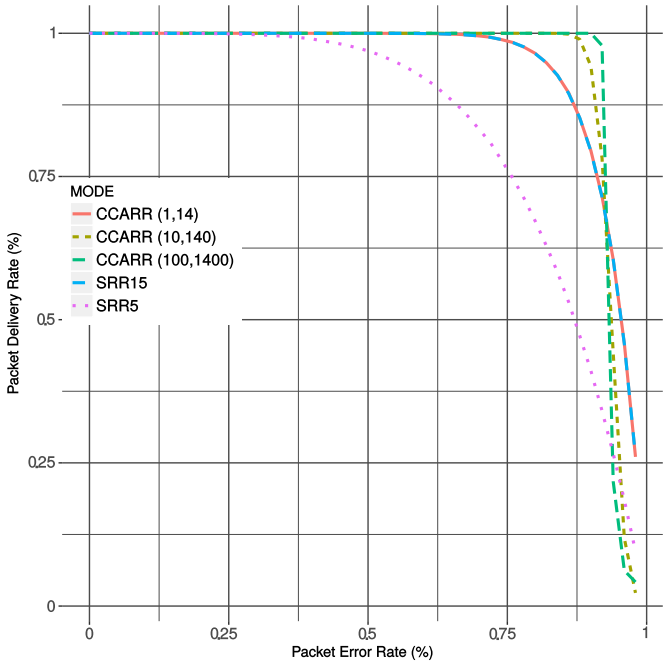

Fig. 8. Average PDR against PER for SRR5, SRR15 and CCARR(a,a×(15-1) protocols, with $a \in \{1, 10, 100\}$.

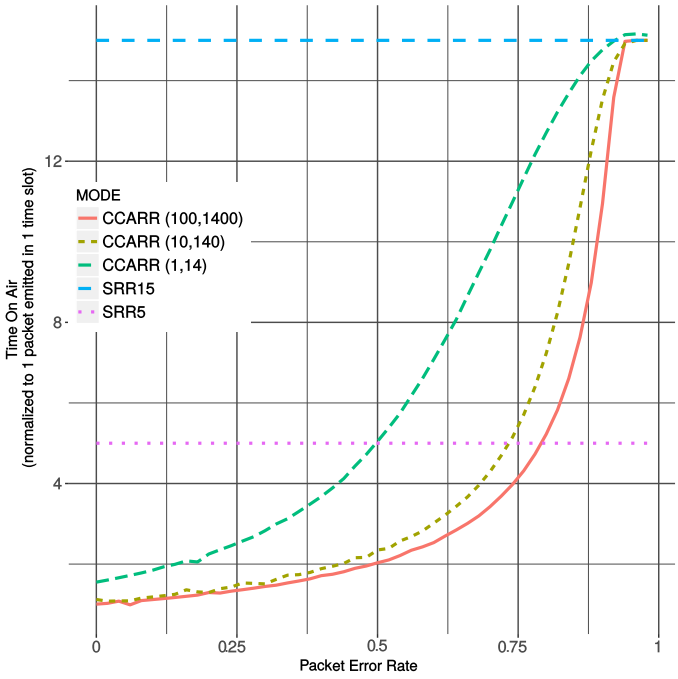

Fig. 9. Average TOA in frame TOA, necessary for data transmission, for the different protocols.

channel. When the channel PER increases, the TOA to emit a frame with CCARR tends to the TOA with the SRR15 protocol. This corresponds to the cases when all the available redundancy frames have to be emitted to recover the data. Notice that, as expected, the CCARR average overload on the channel tends to be close to the minimal overhead that has to be introduced to successfully achieve the communication over a lossy channel. For instance, in order to achieve a 100% PDR over a 50% PER channel, the necessary TOA with CCARR(10,140) is only around 230% of the minimal transmission TOA and CCARR(100,1400) is only a little bit over 200%. Within the simulation assumptions, the CCARR protocol demonstrates a substantial PDR improvement obtained with a low channel TOA.

The comparison with DaRe's PDR isn't trivial since the two FEC schemes are built over totally different mechanisms. As SRR, DaRe uses a fixed FEC amount determined in its case by a priori channel evaluation. CCARR uses a dynamic FEC amount automatically adjusted by the channel PER. Thus, in case of evolving channel, CCARR should provide a better PDR since it automatically adjusts to successfully achieve the transmission. In terms of TOA, the work in [9] provides the following measurements: when DaRe is exactly set to reach 99% PDR on a 10% PER channel, its TOA is 1.3 times the no-FEC transmission TOA while CCARR takes 1.1 on average. This is because DaRe's parameters do not perfectly match channel conditions. Surprisingly, with a 60% PER channel and when the channel conditions perfectly match its parameters, DaRe's 2 times no-FEC TOA reaches 99% of PDR, while CCARR needs 2.6. The reason for this DaRe non-intuitive result[8] might be implementation dependent. Because DaRe's FEC is appended inside existing frames, which save some of the LoRa^{TM} and LoRaWAN^{TM} protocols overhead, while CCARR sends its FEC in independent frames. Anyway, since DaRe do not provide any solution to tune its parameters to match the channel conditions, the best results provided in [9] cannot be reached easily in practice. However, compared to CCARR, DaRe-like solutions do not require any downlink capacity at all.

IX. Experimental Results

Our experimental platform consisted in a gateway station[9] and three end-devices[10] placed in a large room. The gateway was connected to the Internet through wired Ethernet. This Internet connection allowed the access to a network server, TTN[11], which implements LoRaWAN^{TM} specifications. We did not control the behavior of this cloud server. Especially its internal policy where it implements downlink transmissions using a short spreading factor and the maximum downlink allowed power[12]. Hence, this acknowledgement channel was lossless during the experiments. Notice that, compared with the simulation context, it is advantageous for the CCARR protocol to

[8]FEC < the minimum $1/(1-PER)$ needed to fight against PER.
[9]Kerlink Wirnet Station 868 [21].
[10]LoRaMOTE ARM-based end-devices from ISMT [22].
[11]Provided by The Things Networks (TTN) [23].
[12]SF9 and an effective radiated power (ERP) of 27dBm over the 125kHz wide channel centered on 869.525MHz.

TABLE I
Frames TOA during experiments.

Link	SF	TOA (ms)	Payload (Bytes)
Up	12	1646.6	16
Down	9	164.9	2

TABLE II
CCARR implementation extra code sizes[17]

Segment	Text (flash)	BSS(RAM)
Size (Bytes)	6320	5160

get 100% of its completion acknowledgements. With our testbed experimental setup, the acknowledgement's TOA was a tenth of the uplink frame's TOA, as detailed in Table I, while using a large spreading factor to generate contentions. The end-device LoRaMac firmware from [24] was modified to implement the CCARR protocol and its RS-coded FEC. In particular, we modified an RS code library open-source[13] software from [7] in order to fit our embedded software needs in terms of memory management, computation complexity, maximum size of dataword and codeword. The resulting CCARR FEC computation does not constrain the communication system because it takes less than the frame wireless transmission on our end-device MCU[14]. However, even optimized, the RS-code memory footprint is still relatively expensive on our typical MCU[15]. So we limited the segment dimension in order to adjust FEC to typical LoRa^{TM} frames and metering application demand. This resulted in a CCARR(10,140) protocol implantation with twenty-nine Bytes frame for uplinks and 15 Bytes frames for downlinks[16] and a code size fitted to our end-devices as described in Table II.

In order to variate the channel PER, we created various contention levels, by using 1, 2 or 3 end-devices within a single uplink frequency sub-channel[18], at SF12 and an uplink transmission power set to the minimum[19]. Then the desired channel PER was adjusted by tuning the emission rate of the different devices interfering on the contention-based ALOHA channel while respecting the LoRaWAN^{TM} protocol and especially its active listening on receive slots. As a result each end-devices emitted at around 0.25Hz. We monitored the channel PER, the application PDR and the TOA to reach this PDR on uplink and downlink channels. Each experiment was conducted until 150 frames were delivered to the application.

Fig. 10 displays the average PDR reached by the LoRaWAN^{TM} SRR1, SRR5 default protocols, and our

[13]Under GPL license.
[14]32 MHz STM32L151C8U6
[15]10kBytes RAM and 64kBytes flash
[16]Respectively 16 and 2 Bytes of data with 13 control Bytes.
[17]Compiled with arm-none-eabi-gcc version 5.4.1.
[18]868.1MHz.
[19]Two dBm

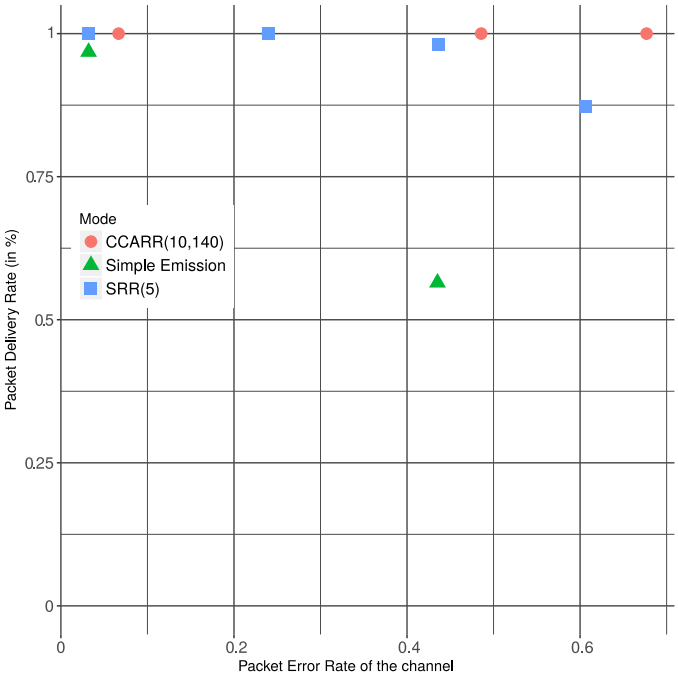

Fig. 10. Average experimental PDR for the different protocols against channel PER.

CCARR(10,140) implementation, against the channel PER. Notice that CCARR(10,140)'s overload in its worst case corresponds to SRR15's one. As expected, the PDR of SRR1 equals the channel quality, ie. (1-PER). The SRR5 measurements have 100% PDR up to 30% PER. Then 98% PDR is obtained for 43% channel PER and 87% PDR for 60% PER. This corresponds to the probabilistic behavior expected in Section VII, as $(0.43)^5 < 0.02$ and $(0.60)^5 < 0.09$. Interestingly, the PDR of the CCARR(10,140) protocol is 100% for all tests up to 67% channel's PER which is the lower stable PER that we were able to reach with our experimental platform.

Fig. 11 displays the measured TOA of the transmissions with the LoRaWAN^{TM} SRR1, SRR5 and CCARR(10,140) protocols, against the channel PER. As in Section VIII, the TOA is in frame TOA unit. As expected, SRR1 and SRR5 emit a constant number of frames which corresponds respectively to one and five uplink frames TOA. CCARR(10,140) TOA increases smoothly with the PER while the protocol compensates the channel degradation. Notice that in these experiments, CCARR corresponds exactly to the number of frame needed to successfully get the data because of the lossless downlink acknowledgement channel. Moreover, CCARR gives 100% PDR for 67% PER channel with an average TOA of 3.3 frames, much less than SRR5.

The experimental conditions with no loss and no downlink acknowledgement interference were favorable for CCARR relatively to the model of Section VIII where up and down links were on the same channel and experienced

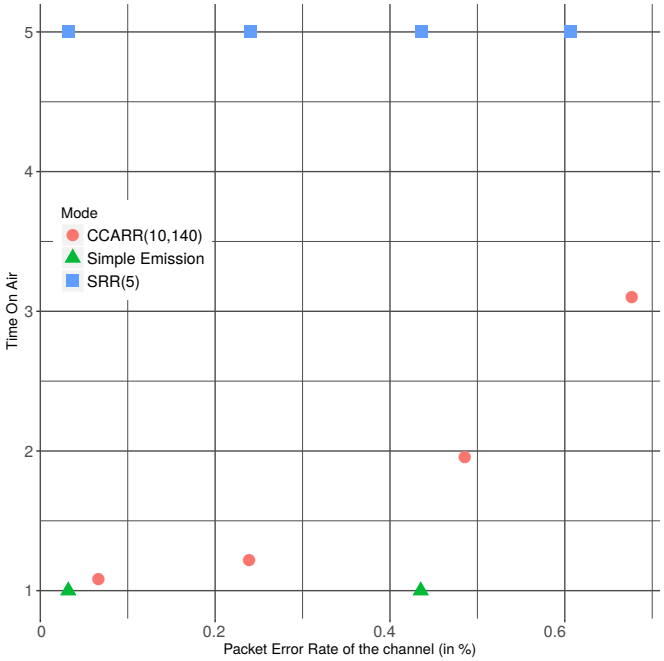

Fig. 11. Average experimental TOA, in frame TOA, for the different protocols against channel PER.

the same PER. However, downlink acknowledgements are usually smaller than regular uplink data payloads, so they may experience smaller channel PER than longer frames on the same wireless link. Beside this, CCARR presents very good practical performance results which corroborates nicely our probabilistic estimations and simulations trends.

X. Conclusion

In this paper we proposed a novel approach that improves LoRa^{TM} LPWAN QoS in terms of application PDR thanks to channel coding. The proposed protocol, Channel Coding Adaptive Redundancy Rate (CCARR) uses Reed Solomon codes FEC. Hence, CCARR provides very high PDR over lossy channels with low overload on the network compared to the existing LoRaWAN^{TM} strategy. As a result, for a given PDR the transmission TOA is smaller with CCARRR. For instance, in our experiments, CCARR(10,140) sustains 100% PDR over a 67% PER channel with only 230% TOA FEC overload. Moreover, the strength of CCARR relatively to SRR and DaRe protocols is its ability to regulate itself according to the effective channel quality. However, the price to pay is seldom segment-based downlink acknowledgments. Nevertheless, CCARR dynamic FEC control is of particular interest for ALOHA access networks, such as LoRa^{TM}. Also it is worth noticing that by reducing the FEC TOA, the CCARR protocol also increases the overall LoRaWAN^{TM} network capacity.

Our future work will assess scalability impacts of acknowledgements, study coding strategies that may answer different application QoS needs and investigate end-devices energy consumption.

References

[1] G. Margelis, R. Piechocki, D. Kaleshi, and P. Thomas. Low throughput networks for the iot: Lessons learned from industrial implementations. In 2nd World Forum on Internet of Things, WF-IoT, pages 181–186. IEEE, 2015.
[2] U. Raza, P. Kulkarni, and M. Sooriyabandara. Low power wide area networks: An overview. IEEE Communications Surveys & Tutorials, 19(2):855–873, 2017.
[3] L.Vangelista, A. Zannella, and M.Zorzi. Long-range iot technologies: the dawn of LoRa^{TM}. In Future Access Enablers for Ubiquitous and Intelligent Infrastructures, pages 51–58. Springer, 2015.
[4] G. Solomon I. S. Reed. Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics, 8(2):300–304, 1960.
[5] D.A. Patterson, G. Gibson, and R.H. Katz. A case for redundant arrays of inexpensive disks (raid). In International Conference on Management of Data, SIGMOD '88, pages 109–116, New York, NY, USA, 1988. ACM.
[6] C. Berrou. Codes et turbocodes. Springer Science & Business Media, 2007.
[7] H. Minsky. sourceforge.net/projects/rscode/, 2009.
[8] Y.Birk and Y.Keren. Judicious use of redundant transmissions in multichannel aloha networks with deadlines. IEEE Journal on Selected Areas in Communications, 17(2):257–269, 1999.
[9] P.J. Marcelis, V.S. Rao, and R. V. Prasad. Dare: Data recovery through application layer coding for lorawan. In Internet-of-Things Design and Implementation, IoTDI, pages 97–108. IEEE/ACM, 2017.
[10] O. Georgiou and U. Raza. Low power wide area network analysis: Can lora scale? IEEE Wireless Communications Letters, 6(2):162–165, 2017.
[11] A.I. Pop, U. Raza, P. Kulkarni, and M. Sooriyabandara. Does bidirectional traffic do more harm than good in lorawan based lpwa networks? arXiv preprint arXiv:1704.04174, 2017.
[12] LoRa Alliance. A technical overview of lora and lorawan. Technical report, LoRa Alliance, 2015.
[13] A. Augustin, J. Yi, T. Clausen, and W.M. Townsley. A study of lora: Long range & low power networks for the internet of things. MDPI Sensors, 16(9), 2016.
[14] N. Abramson. Development of the alohanet. IEEE Trans. Inf. Theor., 31(2):119–123, 2006.
[15] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne. Understanding the limits of lorawan. IEEE Communications Magazine, 55(9):34–40, 2017.
[16] Semtech Corporation. An1200.22 LoRa^{TM} Modulation Basics. Technical report, Semtech, 2015.
[17] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent. Lorawan^{TM} specification. Technical report, LoRa^{TM} Alliance, 2015.
[18] Semtech. SX1272 73 datasheet, 2015. www.semtech.com.
[19] E. Ruano. LoRa^{TM} protocol evaluations, limitations and practical test. Master's thesis, Grenoble INP, ERASMUS from Barcelona, May 2016.
[20] J. Bernoulli. L'art de conjecturer. translated by L.G.F. Vastel and printed by G. Le Roy, Caen, 1801.
[21] Kerlink. Wirnet Station 868, 2016. www.thethingsnetwork.org.
[22] Semtech. LoRaMote User Guide, 2014. www.semtech.com.
[23] Thethingsnetwork. Building a Global Internet od Things Network Together, 2018. www.thethingsnetwork.org.
[24] Semtech. Lorawan endpoint stack implementation and example projects. github.com/Lora-net/LoRaMac-node, 2013.