Journal of Parallel and Distributed Computing 166 (2022) 166–177

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

www.elsevier.com/locate/jpdc

Application-aware adaptive parameter control for LoRaWAN

Ameer Ivoghlian *, Kevin I-Kai Wang, Zoran Salcic

Department of Electrical, Computer, and Software Engineering, The University of Auckland, New Zealand

ARTICLE INFO

Article history:
Received 16 August 2021
Received in revised form 5 April 2022
Accepted 23 April 2022
Available online 2 May 2022

Keywords:
Network dynamics
Large-scale networks
LPWAN
Run-time adaptive

ABSTRACT

Advanced wireless communication technologies are leading towards large-scale, geographically distributed systems, which consist of thousands of co-existing devices with potentially conflicting application requirements, such as high data delivery ratio and low power consumption. At the same time, devices are challenged by varying environmental conditions, which can also lead to degradation in network performance. To ensure the overall network performance and service stability offered by individual devices, the devices must co-exist and adapt to changes in their surrounding environment. In this paper, an application-aware adaptive method is proposed to allow individual devices to dynamically and automatically decide and configure its communication parameters according to its application requirements and environmental conditions. The algorithm aims to enhance and maintain the network performance over time, while allowing every device to satisfy its application requirements. The algorithm is realised on the modern LoRaWAN network protocol. Simulated experiments confirm the proposed algorithm's ability to adapt to changing application requirements at runtime while maintaining network performance. They demonstrate an improvement in packet delivery ratio, energy consumption, and the novel ability to respond to an application's individual performance requirements.

1. Introduction

Wireless sensor networks are rapidly becoming ubiquitous in our homes, across cities, and throughout industries. We interact, directly or indirectly, with tens to hundreds of computing devices every day. Most of these devices, in this Internet of Things (IoT) ecosystem, communicate using wireless radio transmission. Despite the benefits of wireless communication, some issues inherent to wireless networks, such as limited network bandwidth [12], become critical when we scale up. These issues can lead to reduced overall wireless network performance and may greatly impede the growth of the IoT.

The ease of deployment and the reduced infrastructure cost have driven the rapid rise of smart wireless technologies and services. This has been made possible by the development of low power radio and computing technologies. Performance specifications for these wireless technologies do not factor in the varying nature of the physical environment and deployment, which are unpredictable and far from ideal. This changeable nature, which we will term Network Dynamics is a key obstacle to efficient and reliable wireless networks.

Some modern IoT technologies target large-scale, geographically distributed networks. The collection of technologies, which are classified as Low Power Wide Area Networks (LPWANs) [16], are aiming to support a large number of sensing devices. LPWANs are characterised by long range (city wide), low power (multi-year battery life), and low data rate (tens of kbit/s). These characteristics make LPWANs particularly susceptible to the consequences of network dynamics. LoRaWAN, powered by the LoRa radio technology, is one such protocol which promises to support large-scale networks that include thousands of devices (later referred to as nodes) over several kilometres range, with relatively low power consumption. There are however fundamental challenges to realising this scale, especially when faced with network dynamics.

In a well provisioned and stable network, it is reasonable to select communication parameters such that Packet Delivery Ratio (PDR) is 1 to ensure zero data loss, and then optimise other metrics such as power consumption accordingly. However, in situations where there are big variations in network dynamics, striving for a perfect PDR can significantly impact other metrics, such as a significant increase in power consumption. In addition, this is without consideration for the possible co-existence with other nodes with different application requirements. Capturing varying application requirements and facilitating fair access to network resources across multiple applications in a dynamic way is lacking in existing technologies, but will be critical for future IoT ecosystems.

* Corresponding author.
E-mail addresses: ameer.ivoghlian@auckland.ac.nz (A. Ivoghlian), kevin.wang@auckland.ac.nz (K.I.-K. Wang), z.salcic@auckland.ac.nz (Z. Salcic).

https://doi.org/10.1016/j.jpdc.2022.04.023
0743-7315/

To put this into a practical context, consider the following current and future scenarios:

- Smart metering: thousands of nodes, low message rate, external power supply.
- Environmental monitoring: thousands of nodes, higher message rate, battery powered.
- Public network: thousands of nodes, a range of message rates, a variety of power sources.

These scenarios demonstrate that not all networks, or even nodes within a network, have the same application requirements; a gap in existing approaches. They also illustrate the scale of these networks. At this scale, an automated management mechanism is required, which existing approaches address, but the likelihood of congestion also increases, which is often handled poorly.

With a focus on application specific requirements, including multiple applications within a network, node based decision making is most appropriate. It also has the benefit of reduced management overhead. This can be described as a decentralised approach. However, as the primary message receiver, with greater visibility of the network state, base-station decision making can be advantageous. Both centralised and decentralised approaches are presented in related work.

In this paper, we propose a novel application-aware adaptive algorithm that selects the best combination of network parameters according to an individual node's application requirements, which can be in conflict with one another. The proposed algorithm adopts a hybrid *semi-decentralised* approach, where an individual node automatically estimates the best parameters for its application, while receiving occasional assistance from its base station to enhance the quality of its decision. The proposed approach avoids excessive communication overhead compared to a centralised approach where a large amount of control messages would need to be sent to each individual node. At the same time, the approach achieves good performance and outperforms the Adaptive Data Rate (ADR) approach currently implemented in LoRaWAN [4]. The proposed algorithm makes the following contributions:

1. It considers a multi-tenant scenario where a LoRaWAN network consists of devices that belong to several different applications with their own communication requirements.
2. It is adaptive, where individual nodes automatically find the best parameters according to their application requirements, at runtime, based on the current channel conditions.
3. It is a semi-decentralised approach, hosted primarily on the nodes, minimising control messaging overhead.

The remainder of this paper is organised as follows. Section 2 provides a brief overview of existing approaches for network parameter selection. Section 3 explains the design and implementation of the proposed algorithm. The simulation experiments and discussions are presented in section 4, followed by the conclusions in section 5.

2. Related work

This section looks into a range of existing research that investigates ways of finding optimal LoRaWAN network configurations (e.g., spreading factor and transmission power). The first subsection gives a brief overview of various LoRaWAN parameters that can be configured and the associated ADR mechanism, used for more efficient network usage. The next subsection covers other approaches (predominantly ADR-related) for finding optimal LoRaWAN parameters.

2.1. LoRaWAN adaptive data rate mechanism

The low data rate associated with the long range and low power consumption goals of LPWANs demand some form of collision avoidance to realise practical messaging rates at the promised scale. LoRa addresses this by using multiple reception channels at the base station, a range of supported bandwidths, and quasi-orthogonal Spreading Factor (SF) coding within each channel [5][23][14]. Channel frequencies are selected such that there is no overlap between them at a single bandwidth. Base stations generally have the capability of demodulating and processing multiple channels simultaneously. From a network capacity perspective, each channel can be considered independently.

A node's selection of appropriate SF depends on its position within a network and the state of the radio link. LoRa facilitates the automated assigning of SF through the ADR mechanism [4]. ADR does a good job of maximising PDR for a given link environment. However, as demonstrated by Li et al. [13], ADR is best suited to a static and stable link environment and struggles in the face of network dynamics. It also responds poorly to high levels of packet collision, such as those observed in congested networks, as shown by Kim et al. [11].

To illustrate this point we can consider how ADR would behave in a congested network. Nodes with enough link margin to reach the base station, but not enough to dominate an overlapping packet on the same channel and spreading factor, would experience some packet errors. In the absence of any collision detection, ADR would react to these missed packets by attempting to increase the link margin by increasing transmission power or spreading factor. Increasing transmission power raises the average noise on the channel. Increasing spreading factor increases packet time-on-air, both aggravating the congestion. The positive feedback loop which is created here can potentially cripple the network.

2.2. Other mechanisms

Li et al. [13] investigate the ADR mechanism and evaluates its performance under changing link conditions. They measure how quickly a node converges on the correct spreading factor and transmission power settings when faced with a change in link quality. Their simulations utilise LoRaWANSim [15], an extension of LoRaSim [3]. Their results demonstrate that ADR convergence time has a strong correlation with both network size and link quality degradation. The authors claim that the ADR mechanism lacks the agility to adapt to changing link conditions, requiring a number of hours and even days to converge to a reliable and energy-efficient communication state. In response to this, they investigate how modifying some of ADR's tunable parameters can improve convergence time. They also propose a direction for future research.

Similarly, Slabicki et al. [22] demonstrate that while ADR is effective in improving PDR under stable channel conditions, it is severely hindered when the wireless channel has high variability. They propose ADR+, an algorithm replacing ADR, which performs well on a highly variable channel. The primary change is to replace ADR's estimate of link quality (highest Signal to Noise Ratio (SNR) over the last 20 frames) with a more conservative approach (average SNR over last 20 frames). Clearly, in a stable link scenario, the average SNR will be similar to the highest SNR, and would thus provide a similar level of performance. In an unstable link scenario however, the more conservative approach provides more headroom to cope with difficult conditions.

Hauser et al. [9] investigate the performance of ADR and suggests a range of changes. Their first proposed change is to decrement data rate before incrementing Transmission Power (TXP). This protects the node from becoming stuck in a configuration where it cannot communicate with the base station. Their second

Table 1
Comparison of Related Works.

Ref	Name	Approach	Parameters	Significant Results
[13]		Tuning ADR parameters to improve convergence time.	nodes: 4000, range: 670 m, period: 10 m	32% reduction in convergence time and 23% reduction in energy consumption when ADR_ACK_DELAY reduced from 32 to 16.
[22]	ADR+	Modify ADRs estimate of link quality from max SNR to average SNR.	nodes: 700, range: 679 m, message: 20 B, period: 16.7 m	150% improvement in PDR and 47% reduction in energy consumption for ADR+ relative to ADR, with typical path loss variance.
[9]		Decrement DR before incrementing TXP at ADR server.		30% increase in PDR and 70% increase in power consumption relative to ADR when path loss is significant.
[2]	FADR	Use TXP to equalise RSSI between nodes, making SF independent of distance.	nodes: 1000, range: 1100 m, message: 80 B, period: 60 s	Consistent DER performance for nodes between 150 m and 750 m. Markedly more stable than the approach in [17].
[6]	EXPLoRa-AT	Distributes nodes between SFs to equalise time-on-air and take advantage of SF orthogonality.	nodes: 2000, range: 100 m, message: 20 B, period: 60 s	67% increase in DER and 63% increase in throughput relative to ADR.
[11]		Increases node back-off time instead of adjusting SF if congested.	message: 100 B	39% reduction in transmission delay relative to ADR.
[18]	approx-alg	Minimise utilisation of CF and SF pairs, reducing chance of collision.	nodes: 1500, range: 99 m, message: 20 B, period: 16.6 m	58% increase in DER and 63% decrease in energy consumption relative to ADR.
[7]		Use payload replication with compression to improve data delivery.	range: 100-1000 m, period: 1-10 m	Data delivery ratio increases by 45% relative to using no replication.

proposed change is to utilise an exponentially decaying weighted average of the past N samples of SNR. This is similar to the approach proposed by Slabicki et al. [22], albeit more computationally intensive. Their third proposal is to introduce hysteresis into the ADR algorithm so that nodes don't unnecessarily oscillate between two configurations when the hypothetical ideal configuration sits somewhere in between. They present some limited results from simulations which illustrate the performance of all possible combinations of the three proposed changes against standard ADR. All configurations appear to provide improvements in packet delivery and energy consumption compared to ADR.

Abdelfadeel et al. [2] begin by deriving a data rate distribution which achieves a fair collision probability among all nodes in a LoRaWAN network. They then develop a transmission power control algorithm *FADR*, based on this distribution, which adjusts TXP to equalise the RSSI between all nodes, regardless of distance from the base station. By equalising the RSSI between nodes, the occurrence of the capture effect is reduced, making the choice of SF independent of the node's distance from the base station, which the authors claim is *fairer*. Using LoRaSim [3], they show that *FADR* achieves an almost uniform data extraction rate (DER) from all nodes, independent of distance.

Cuomo et al. [6] propose two approaches for allocating spreading factors to nodes. The first, EXPLoRa-SF, equally distributes nodes across all spreading factors (RSSI permitting). This takes advantage of LoRa's spreading factor orthogonality, a quality that ADR does not have. The second approach, EXPLoRa-AP, extends this concept by equalising the time-on-air. Higher spreading factors logically use more time-on-air, and distributing nodes evenly without considering this would unevenly load higher spreading factors. Simulation results demonstrate a 41% and 67% improvement in DER, relative to ADR, for EXPLoRa-SF and EXPLoRa-AT respectively.

Kim et al. [11] identify that ADR is poorly suited to congested network scenarios. ADR is not congestion aware and will in fact exacerbate the problem. The authors propose addressing this with a mechanism composed of a congestion classifier and a data rate controller. The congestion classifier estimates whether the network is congested (true/false) using throughput, RSSI, and connection count, as inputs. If the congestion classifier decides that the network is congested, it advises nodes to adjust their back-off time. Otherwise, it advises them to adjust their data rate. The authors claim that congestion is not fixed by pure data rate control, and that when congestion occurs, a back-off procedure is more appropriate. The results demonstrate a reduction in transmission delay for their approach, the benefits becoming significant as packet count increases. They note that because of the high computational cost, the congestion classifier is currently located centrally on the base station, with future work investigating the possibility of moving it onto the nodes.

Sallum et al. [18] seek to increase the performance of LoRaWAN networks through the selection of radio parameters. They tackle this as an optimisation problem using mixed integer linear programming (MILP) to find optimal SF and Carrier Frequency (CF). Their MILP problem (which uses an approximation algorithm) seeks to minimise the utilisation of CF and SF pairs, thereby minimising the chance of collisions. This assumes SF orthogonality. The selection of SF and CF is done at the application layer, on the application server, and it is transmitted to the nodes using a modified base station, thereby keeping the nodes compatible with the standard. By considering the network traffic specifications as a whole, they improve the DER by an average of 6.6%, collisions reduced by 13.3 times, and energy consumption increased by 2.9 times compared to standard LoRaWAN.

Dix-Matthews et al. [7] bring to attention the reliability and energy trade-off, more specifically data-delivery-ratio versus energy-consumption-per-packet. As has been demonstrated by this short literature review, focus is often given to maximising PDR, with reducing energy consumption a secondary objective. The authors of this paper suggest that high rates of packet delivery can be achieved, even when optimising energy consumption. They demonstrate that the selection of energy conservative LoRa radio parameters, combined with forward-error-correction and payload replication, can improve data delivery by 16-25%, while maintaining a similar level of energy consumption, with future work attempting to demonstrate this with a range of real scenarios.

Table 1 presents the most significant results from related works.

2.3. Application awareness

It is difficult to find related works for run-time application awareness in the context of wireless sensor networks. There is however a significant body of work on multi-objective optimisation. The survey paper by Fei et al. [8] identifies coverage, delay, node count, bit-error rate, and network lifetime as the most commonly used metrics when considering wireless sensor network Quality of Service. They categorise various approaches for solving these multi-objective optimisation problems. Some of the most relevant approaches include:

- Mathematical programming methods (e.g. linear weighted sum, goal programming).
- Nature-inspired metaheuristic algorithms (e.g. genetic algorithms, reinforcement learning).
- Other advanced techniques (e.g. fuzzy logic, game theory).

They also aptly highlight that many metrics conflict with each other, and that there is a natural trade-off. This trade-off must be resolved by some form of objective function. The metric-conflict feature is also highlighted in the survey paper by Iqbal et al. [10]. This paper concludes that it is desirable for the optimisation algorithm to be implemented on the node itself. Our adaptive, application aware approach addresses this need.

Compared to the aforementioned approaches, our adaptive parameter control algorithm differs in the following aspects.

1. To the best of our knowledge, this is the first algorithm that is application aware, and considers a multi-tenant scenario where a LoRaWAN network consists of devices with multiple applications, with different network requirements.
2. Our proposed approach is a semi-decentralised adaptive algorithm, where individual nodes automatically find the best operating parameters according to their application requirements at runtime. The base station only provides occasional corrections while complying with duty cycle constraints.

The details of the algorithm are presented in the next section.

3. AAPC: application-aware adaptive parameter control

This section introduces the design of the application aware adaptive parameter control (AAPC) algorithm. The first two subsections describe the adjustable network parameters that will be configured by the algorithm, followed by the description of metrics used to evaluate the network performance in order to adaptively update the parameters according to the current communication environment. The third subsection provides the top level architecture of AAPC, then followed by two subsections describing individual components of AAPC.

3.1. Adjustable parameters

In a fully decentralised system, each individual node will monitor its own performance metrics and then pick the best set of parameters. In practice, this can be difficult to achieve, because some metrics such as the Packet Delivery Ratio (PDR) can only be measured at a base station. Such information is not available to a node without feedback or coordination from the base station. Regular exchange of information between a node and its base station (i.e., a centralised approach) enables more accurate measurement of node performance, but may introduce significant communication overhead. Such overhead is not acceptable in LPWANs as has already been stated by Wu et al. [24].

On that account, we therefore propose a hybrid "semi-decentralised" approach to allow individual nodes to adaptively control their own parameters, with limited assistance from their base station. Our method requires that a node estimates its own performance using adopted metrics. The node regularly communicates its estimates to the base station inside data packets, thereby reducing the overhead introduced by special control packets. Meanwhile, the base station keeps its own measure of node's performance. When a node estimate is received and significantly deviates from base station's observation, the base station will then issue a downstream correction to the node. The trade-off between control accuracy and control-message overhead can be tuned in practice according to various network constraints, such as the duty cycle limitation in LoRaWAN.

In AAPC, the following two parameters are selected to control and optimise a LoRaWAN network's performance.

3.1.1. Spreading factor (SF)

LoRa supports quasi-orthogonal [14] SFs. This enables the decoding of multiple simultaneous transmissions at the same frequency, as long as they use different SFs and similar receiving power levels. Each step of increase in SF results in an increase in time-on-air (ToA), or a reduction of data rate, but it provides the benefit of 2.5 dB of extra link budget. Moving from SF7 to SF12 provides 12.5 dB of link budget. That corresponds to approximately 4 times increase in maximum transmission range in free-space. However, it also decreases data-rate, and therefore increases ToA, by a factor of 18 times. Adjusting this parameter allows us to exploit the trade-off between data rate and link budget.

3.1.2. Transmission power (TXP)

The main role of TXP control in wireless communication is to affect the link budget. For a packet to be successfully decoded, it must arrive at the receiver with a signal level greater than the receiver's effective sensitivity (plus a link margin, if any). LoRa transceivers support a range of configurable TXP levels. The Semtech SX1276 [21] used for this research supports a range of TXP values. AAPC uses values between -4 dBm and 14 dBm in 2 dB steps. Increasing TXP from -4 dBm to 14 dBm results in 18 dB of extra link-margin, or approximately 8 times increase in maximum transmission range in free-space. Correspondingly, the energy consumption of the transmitting device increases by a similar factor. In the case of packet collision, higher TXP increases the chances of one transmission dominating another by way of the capture effect. It is clear that TXP will have a profound effect on network performance and it is sensible to include this trade-off in our evaluation.

3.2. Performance metrics

In this paper, the terms node performance and network performance have both been used, and it is important to make a distinction between them. Because of the semi-decentralised nature of the approach, and the focus on application-awareness, the target is on improving node performance. Node performance reflects the application specific requirements of each node. Network performance, however, is captured using metrics which reflect the overall performance of the network as a whole, and it may conceal the behaviour of individual nodes. AAPC operates on node performance metrics, but the evaluation in this paper averages the performance of all nodes, which is a network performance metric.

Performance is evaluated with the following two metrics according to the configured parameters.

3.2.1. Packet delivery ratio (PDR)

PDR is a key measure of node performance. It is influenced by a range of factors, from radio transmission and reception, to network layer effects. It captures the impacts of these factors and provides an indication to the overall communication performance. Specifically, the PDR is defined in (1), as the ratio of successfully delivered packets to total packets sent.

$$\mathrm{PDR} = \frac{packets\ delivered}{packets\ sent} \tag{1}$$

Because PDR is computed based on the proportion of packets which arrive correctly at the base station, it is difficult to estimate it on the transmitting node without feedback from the receiving base station. This feedback often takes the form of packet acknowledgements, but as discussed at the beginning of this section,

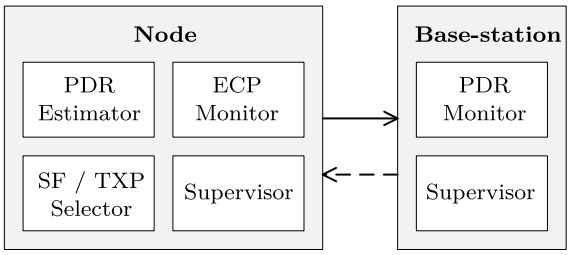


Fig. 1. Top Level components used in algorithm implementation.

this mechanism is not appropriate for large-scale, low data-rate networks with limited bandwidth. In contrast, PDR can easily be measured on the base station utilising packet counters. If the node numbers each packet sequentially, as commonly done by many protocols, missing packets can be detected, and PDR quantified.

3.2.2. Energy consumption

Energy consumption is an inescapable design constraint in battery powered wireless devices and applications. Battery replacement during a node's lifetime can be a significant portion of the network maintenance cost, so minimising how often this happens, by reducing energy consumption, can be beneficial. Therefore, energy consumption is considered another key performance metric according to different application requirements.

There are several ways of measuring network energy consumption. In this research, the energy consumption per packet (ECP), as defined in (2), is used because it is application agnostic, which suits the goal of satisfying different application requirements.

$$\mathrm{ECP} = V_{supply} \times I_{supply} \times T_{packet} \tag{2}$$

Where V_{supply} is the supply voltage, typically a constant value from a regulated battery source; I_{supply} is the current draw based on V_{supply} and the configured TXP; T_{packet} is the transmission time of a packet.

3.3. Top level architecture

The main objective of AAPC is to find the set of parameters (i.e., SF and TXP) that will maximise a node's communication performance according to its application requirements. Fig. 1 illustrates the core components of the algorithm on a node and the base station. Generally, the algorithm works as follows: The node component estimates node performance using the defined metrics and uses these values to calculate an overall performance index formulated as a weighted function (refer to Section 3.5.3). It also sends these estimates to its base station within standard data packets. The base station, at the same time, measures the node's performance using the same metrics and compares these to node estimates. Necessary corrections, if any, are sent downstream in special control packets. The correction information is used by the node to adjust and improve its estimate, and hence its next performance index calculation. The pair of parameters which yield the best performance index is used, which is expected to best meet the application performance requirements.

3.4. Base station components

In the context of AAPC, the base station's objective is to monitor a node's performance metric estimations and ensure they do not deviate from its own (true) observation. Considering the two performance metrics that have been selected, PDR can only be accurately calculated at the base station, whereas ECP can only be accurately calculated on a node. Thus, the base station's task

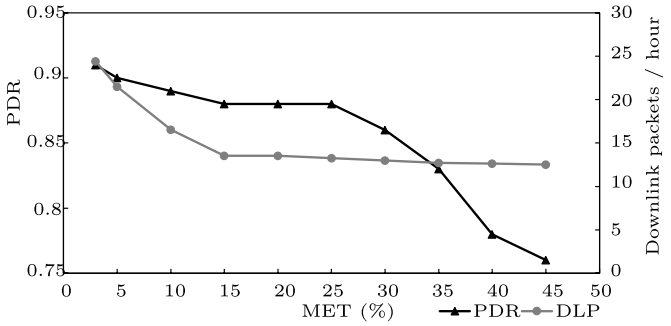


Fig. 2. Effect of MET on PDR and Downlink Correction Messages.

of correcting node's estimates only applies to PDR. Therefore, as shown in Fig. 1, there are two base station components, PDR Monitor and Supervisor.

3.4.1. PDR monitor

The PDR monitor observes and computes the PDR at the base station according to (1), where packets-sent is calculated using a packet counter inside the packet. Algorithm 1 describes this process.

Algorithm 1: Base-station: PDR Calculation.

Data: node ID: n, PDR monitoring buffer: $FIFO_n$, buffer index: X_n
Result: PDR value

$F \leftarrow FIFO_n$
$L \leftarrow length(F)$
$x \leftarrow X_n$
$M \leftarrow packetCnt_n - packetCntPrev_n - 1$
for $i \leftarrow 0$ ***to*** M **do**
  $F_x \leftarrow 0$
  $x \leftarrow (x+1) mod(L)$
$F_x \leftarrow 1$
$x \leftarrow (x+1) mod(L)$
$PDR \leftarrow \left(\sum_{i=0}^{L} F_i\right)/L$
return PDR

3.4.2. Supervisor

The supervisor's primary task is to take the values computed by a metric monitor (in this case, the PDR Monitor, which can be further generalised into multiple monitors), and compare them to the estimates provided by the nodes. If the node's estimate and base station calculation differ by more than the Metric Error Threshold (MET), a correction must be sent to the node. The MET assumes the same unit as its associated metric. In the case of PDR, a MET of 0.05 means that the base-station calculation of PDR and the node's estimate can only differ by 5% before a correction must be issued. Fig. 2 demonstrates the effect of changing MET on both PDR and downlink correction packets (DLP), for a 128 node, single channel network. At present, MET is set to 15%, which maximises PDR precision without significantly increasing base station to node messaging. Further research is needed to ascertain if the current MET value is suitable for all scenarios and environments, or if it can be generalised.

3.5. Node components

The objective of the node (and its associated components) is to find the parameter set which best fulfils the application requirements. Fig. 3 illustrates the four node components, namely PDR Estimator, ECP Monitor, SF/TXP Selector, and Supervisor, which all execute concurrently and communicate with each other. These components are discussed in detail in the following subsections.

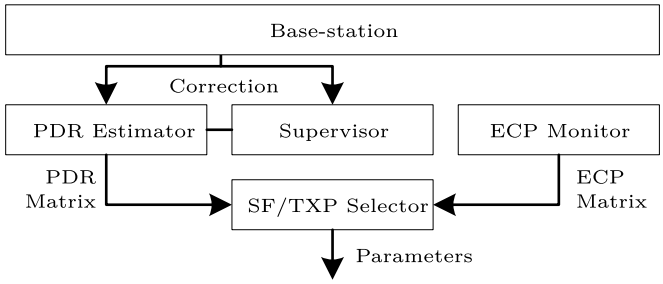


Fig. 3. Node level algorithm illustration.

3.5.1. PDR estimator

Without a packet delivery acknowledgement, there is no direct way for a node to measure its PDR. It can however, make local estimates and rely on base station feedback to correct errors. These estimated and corrected PDR values are stored in a matrix, where each element corresponds to a combination of adjustable parameters SF and TXP. Initially, the node will wait for the base station to provide two PDR values. Given two reliable PDR values in the matrix, the node can then estimate by interpolating/extrapolating PDR values for the rest of the matrix.

It is obvious that a recent base station PDR correction should not be overwritten by an interpolated or extrapolated value, in which we have less confidence. However, we are targeting systems under the influence of network dynamics, and we would therefore expect the received PDR values and local estimates to drop in accuracy over time. Our confidence in all values, corrected or interpolated, must be reduced as time progresses. This is handled within the PDR Estimator by assigning a confidence index to every PDR value; 100 for base station corrections, 90 for interpolations, and 50 for extrapolations. Interpolations are estimated in-between two known data points, which are assumed to have higher confidence, whereas extrapolated ones have lower confidence. PDR values can only be overwritten by a new value if the source has a higher confidence index. This process is described in Algorithm 2, which is executed whenever a correction is received from the base-station.

Algorithm 2: Node: PDR Matrix Update.

Data: $confidence$ matrix, $correction$ from base-station
Result: PDR matrix
if $correction = 0$ **then**
  $PDR_{SF,TXP} \leftarrow 0$
  $confidence_(SF, TXP) \leftarrow 50$
  $PDR \leftarrow 0$ for all lower SFs and TXPs
  $confidence \leftarrow 0$ for all lower SFs and TXPs
else
  $PDR_{SF,TXP} \leftarrow correction$
  $confidence[SF][TXP] \leftarrow 100$
  Find higher SF value with confidence > 90
  Interpolate intermediary values using gradient
  Assign confidence value 90 for interpolations
  Extrapolate beyond that
  Assign confidence value 50 for extrapolations
  Repeat for lower SF, lower TXP, higher TXP

The confidence indices for all parameter sets are decremented every hour. In effect, confidence in a corrected value drops low enough as to be overwritten by an interpolated value in 10 hours, or an extrapolated value in 50 hours (about 2 days). Like MET (section 3.4.2), or PDR measurement resolution (section 4.7), the rate at which confidence is decremented can be tuned to control a trade-off. In this case, the trade-off between responsiveness and stability. Further research is needed to determine if this feature can be generalised.

3.5.2. ECP monitor

Unlike PDR, ECP can be directly computed by a node itself. Using (2), an ECP matrix (similar to PDR, with rows and columns corresponding to SF and TXP combinations) can be computed. The computation of ECP matrix is detailed in Algorithm 3. This procedure is executed at node initialisation, or when the payload length changes. It is worth noting that the computed ECP value is also normalised such that it is in the same range as PDR, which can later be combined by a weighted function to compute a performance index. Specific to LoRaWAN, the packet transmission time, T_{packet}, can be calculated using (3)-(6) [19].

Algorithm 3: Node: ECP Matrix Update.

Data: voltage: V_{supply}, supply current list: I, preamble symbols: N_{pream}, payload length (B): N_{pl}, bandwidth: B, FEC coding rate: C, implicit header: H, low data-rate optimisation: D
Result: normalised ECP matrix
for $i \leftarrow 7$ ***to*** 12 **do** // SF7 to SF12
  $T_{packet} \leftarrow$ $\left(\frac{2^i}{B}\right) \times \left(N_{pream} + 12.25 + \max\left(\left\lceil \left(\frac{8N_{pl}-4i+44-20H}{4(i-2D)}\right)\right\rceil \times (C+4), 0\right)\right)$
  for $j \leftarrow -4$ ***to*** 14 **do** // -4 to 14 dB
    $I_{supply} \leftarrow I_j$
    $ecpRaw_{i,j} \leftarrow V_{supply} \times I_{supply} \times T_{packet}$
for $i \leftarrow 7$ ***to*** 12 **do**
  for $j \leftarrow -4$ ***to*** 14 **do**
    $ECP_{i,j} \leftarrow \frac{ecpRaw_{i,j} - \min(ecpRaw)}{\max(ecpRaw) - \min(ecpRaw)}$

The transmission time of an individual symbol can be computed using spreading factor and bandwidth as defined in (3). The transmission time for preamble and data payload can then be computed based on (4)-(5):

$$T_{sym} = \frac{2^{SF}}{B} \tag{3}$$

$$T_{pream} = (N_{pream} + 4.25)T_{sym} \tag{4}$$

$$T_{payload} = \left(8 + \max\left(\left\lceil \frac{8N_{pl} - 4SF + 44 - 20H}{4(SF - 2D)} \right\rceil \times (C+4), 0\right)\right)T_{sym} \tag{5}$$

where:

SF = spreading factor
B = bandwidth (kHz)
N_{pream} = number of configured preamble symbols
N_{pl} = number of payload bytes
H = header disabled (bool)
D = low data-rate optimisation enabled (bool)
C = coding rate (1 to 4)

The overall packet transmission time is defined in (6):

$$T_{packet} = T_{pream} + T_{payload} \tag{6}$$

3.5.3. SF/TXP selector

With the PDR and ECP matrices fully populated and independently updated, the SF/TXP Selector component can compute a performance index for every parameter combination, creating a combined performance matrix. The maximum value within that matrix indicates the TXP and SF combination that best matches the

application requirements at that point in time. The performance index is formulated as a weighted function in the following form:

$$\rho = \beta_1\omega_1 M_1 + \beta_2\omega_2 M_2 + \ldots + \beta_n\omega_n M_n \tag{7}$$

where

$$\sum_{i=1}^{n} \omega_i = 1 \tag{8}$$

and

$$\beta_i = \begin{cases} U_i/M_i, & M_i > U_i \\ 1, & U_i \geq M_i \geq L_i \\ 0, & M_i < L_i \end{cases} \tag{9}$$

M_i is the value of performance metric i for the parameter set being evaluated. ω_i is the relative weight of that metric with respect to the others, selected to reflect the application requirements. Maximising ρ reveals the configuration which best reflects the requirements. β_i captures any limits that might apply to the metrics, such as a minimum acceptable value or lower limit L.

The first case in (9) handles metric values which exceed the specified upper limit U, by ensuring that no further gains are achieved beyond this limit. The second case handles metric values which fall within the specified upper and lower limits, and leaves the value unchanged. The third case handles metric values which are below the lower limit L, and applies a hard penalty on these values by cancelling any contribution they could make to the objective function.

This case statement can be modified to soften the limits, by instead applying a more modest penalty to values below L, or providing diminishing returns for values above U, however this has not been implemented in this study.

3.5.4. Supervisor

Nodes have the potential to get stuck in a sub-optimal condition. Network dynamics will change the communication environment, but the node is not directly aware of this. With the information it has on hand, it assumes that its current parameter set values are the most appropriate. As discussed in 3.5.1, the time-based update mechanism within the PDR Estimator reduces all PDR values uniformly. This approach alone cannot compensate for network dynamics. The supervisor encourages the nodes to test parameter sets which appear unfavourable, to see if they have improved relative to the other sets. This is achieved by periodically promoting the PDR value of low ranking parameter sets so that the SF / TXP Selector naturally selects and tests them. How aggressively this is carried out affects the trade-off between system responsiveness to network dynamics and the overhead of testing potentially unfavourable parameter sets.

4. Evaluation

As mentioned in the previous sections, AAPC aims to support large-scale and geographically distributed LoRaWAN networks, which are highly likely to host thousands of devices with varying application requirements. It is therefore more feasible to conduct simulated experiments instead of physical deployment. In this section, we first introduce a purpose-built modelling framework which has been developed to assist in this endeavour, followed by a few different scenarios of simulation experiments to evaluate the performance of AAPC.

For consistency, the performance of AAPC is always compared to ADR which is also simulated in the modelling framework. ADR has been enabled continuously to maximise its responsiveness.

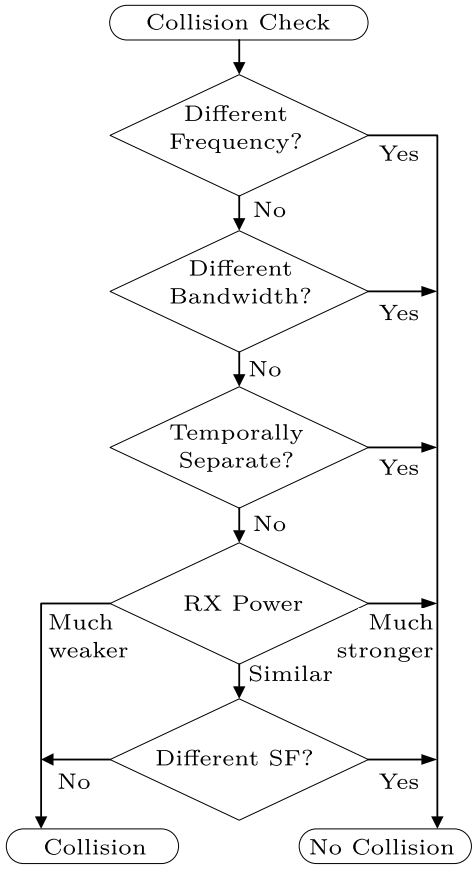


Fig. 4. Collision detection process between two received packets.

Furthermore, our implementation of ADR uses a fixed link margin of 10 dB, as done in [13], instead of Semtech's recommended 5 dB which proved inadequate in our high path-loss scenarios and results in severely reduced performance for ADR. Comparisons between AAPC and related works can be made using the summary in Table 1, as many of these works have also been evaluated against ADR.

4.1. Simulation framework

Inspired by the work in [3], our modelling framework has been designed using SimPy [1]. It facilitates the execution of many independent and concurrent processes while simplifying inter-process communication. It is a convenient platform to model or simulate the behaviour of a communication network, especially asynchronous ones with a low level of coordination such as LoRaWAN. The base station and each node are modelled as independent processes, and executed concurrently.

A Log Distance path loss model is used which incorporates the shadow fading experienced in denser environments. Adjusting the path loss exponent, and standard deviation of the normally distributed shadow fading accommodates the simulation of different transmission environments. The scenarios simulated in this evaluation take place in a dense, urban environment, with high path loss and variability.

On top of this, the LoRa specifications (e.g. coding rate and packet delivery time in (3)-(6)) are also implemented following the Semtech LoRa Application Notes [19][20]. The developed simulation framework allows link margin to be evaluated per packet, and packet collision to be tested by comparing packets for interaction, as shown in Fig. 4.

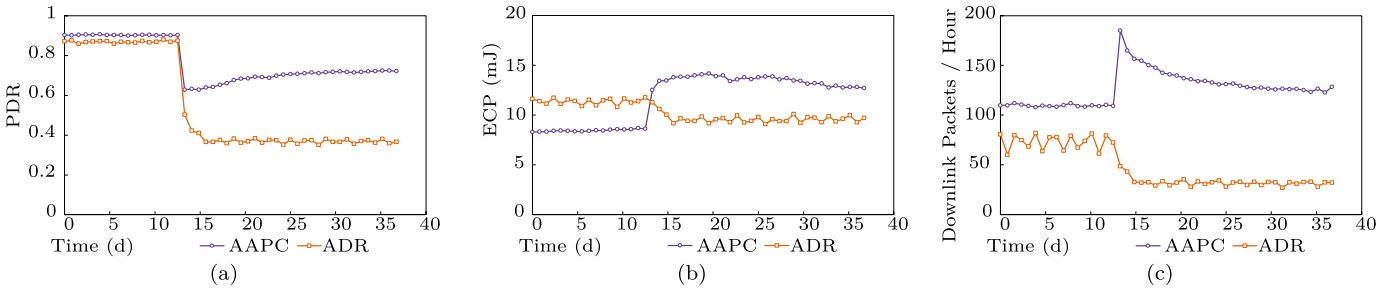


Fig. 5. (a) Mean network PDR for scenario 1; (b) Mean network ECP for scenario 1; (c) Downlink packet count for scenario 1.

Table 2
Simulation Parameters.

Parameter	Value
Frequency Band	915 MHz
Uplink Channel Bandwidth	125 kHz
Uplink Channel Spacing	200 kHz
Uplink Channels	8
Base stations	1
Cell Radius	750 m
Nodes	1024
Node Distribution	Random Uniform
Spreading Factors	SF7 .. SF12
Transmission Power Range	-4 dBm .. 14 dBm
Transmission Power Steps	2 dB
Coding Rate	4/5
Message Period	Variable
Payload Length (PL)	Variable
Frame Length	PL + 148 bits
ARQ	Off
ADR Margin	10 dB
Path Loss Exponent (λ)	3.5
Shadow Fading Std Dev (σ)	9

The base station model consists of a radio which can receive on 8 channels together with another which can independently transmit on 1. This allows for simultaneous uplink and downlink messaging.

4.2. Simulation parameters

Table 2 lists the parameters which are common to all the results presented in this section. Payload Length and Message Period are scenario specific and their values can be found in the following subsections.

For both ADR and AAPC, all nodes are initialised to SF12 and TXP14 dB to maximise the probability of successful communication with the base station.

4.3. Scenario 1: smart meters

Scenario 1 simulates the communication between smart power meters and a gateway. The message period is 30 minutes and the payload length is 12 Bytes. Given that a smart power meter is unlikely to be battery powered, this application gives priority to PDR over ECP, and therefore uses the following metric weights: $\omega P = 0.9$, $\omega E = 0.1$. This simulation evaluates AAPC's ability to respond to network dynamics. In this case, a co-located network, utilising the same radio band, is deployed 12 days into the simulation. This is represented as significant noise with a mean of 6 dB and log normal variance with a standard deviation of 6 dB. The results of this simulation are presented in Fig. 5a, 5b, and 5c.

As illustrated in Fig. 5a, both AAPC and ADR have similar average PDR when no interference is present. However, when impacted by interference, ADR drops significantly more than AAPC does. Fig. 5b illustrates the energy consumption per packet (ECP), and we can see that AAPC responds to this interference by increasing SF, TXP, or both. This results in an increase in average power consumption. We would expect the ADR algorithm to respond in a similar way, but we can see from Fig. 5b that ECP in fact drops. Given the variance in the introduced interference, it is likely that ADR's use of the $maxSNR$ metric, rather than an average SNR, is responsible for the sub-optimal selection of SF and TXP parameters. This is supported by the findings in [22]. We can see from Fig. 5c that AAPC's downlink messaging is significantly higher than ADRs. This directly represents AAPCs overhead as it does not use special uplink packets. However, at its peak, it equates to just 23 packets per hour, per channel, which consumes less than 0.2% of airtime using the worst case SF12.

4.4. Scenario 2: environmental monitoring

Scenario 2 simulates an environmental monitoring network. The message period is 10 minutes and the payload length is 8 Bytes. These nodes are typically battery powered, and so this application prioritises ECP using the following metric weights: $\omega P = 0.2$, $\omega E = 0.8$. This simulation evaluates AAPC's response to a change in application requirements. In the context of the environmental monitoring scenario; 19 days into the simulation, 10% of the nodes change their application requirements to prioritise PDR in response to a local event, such as an environmental variable exceeding an alarm threshold. This change in application requirement is captured by the following metric weights: $\omega P = 0.99$, $\omega E = 0.01$. The results of this simulation are presented in Fig. 6a, 6b, and 6c.

We can see in Fig. 6a and 6b, that the 10% of nodes reacting to the event, and the 90% that are not (denoted App1 and App2 respectively), initially favour energy consumption. When the local event occurs, App1 nodes experience a change in PDR and ECP levels, settling near ADR's values, but with a slightly higher PDR and lower ECP. The local event also corresponds with a surge in downlink messages as illustrated in Fig. 6c. Similar to scenario 1, this equates to less than 0.4% of airtime using the worst case SF12.

4.5. Scenario 3: public network

Scenario 3 simulates a public network, consisting of multiple, unrelated applications. It extends the previous scenario to evaluate AAPC's ability to handle multiple application requirements simultaneously. The network contains three applications, denoted App1, App2, and App3. They use the following metric weights: [$\omega P = 0.9$, $\omega E = 0.1$], [$\omega P = 0.6$, $\omega E = 0.4$], and [$\omega P = 0.2$, $\omega E = 0.8$]. While in reality, different applications would use different message periods and payload lengths, for the sake of this evaluation some variables must be controlled. Therefore, a message period of 8 minutes and a payload length of 20 Bytes is used by all applications. The results of this simulation are presented in Fig. 7a, 7b, and 7c.

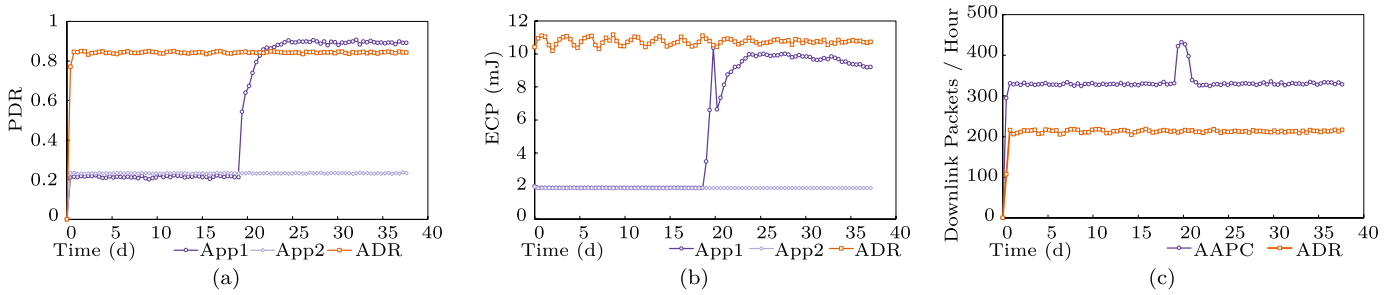


Fig. 6. (a) Mean network PDR for scenario 2; (b) Mean network ECP for scenario 2; (c) Downlink packet count for scenario 2.

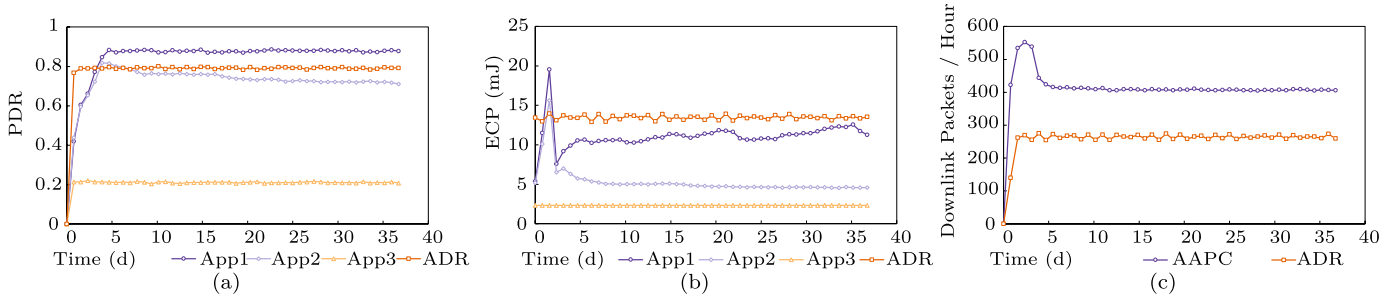


Fig. 7. (a) Mean network PDR for scenario 3; (b) Mean network ECP for scenario 3; (c) Downlink packet count for scenario 3.

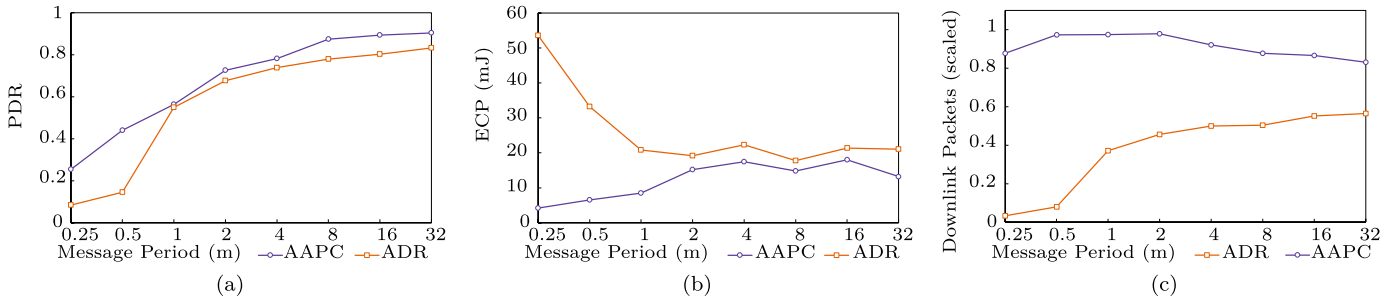


Fig. 8. (a) Mean network PDR against message period; (b) Mean network ECP against message period; (c) Downlink packet count against message period.

As Fig. 7a and 7b demonstrate, each application's requirements are being addressed simultaneously. App1 is clearly PDR focussed, App3 ECP focussed, and App2 somewhere in-between with a PDR lean.

4.6. Congestion response

This series of simulations seek to evaluate AAPC's response to congestion. However, instead of carrying out a single scenario, AAPC and ADR have been evaluated across a range of message periods, and their responses plotted. A PDR priority application has been selected, aligning its goal with ADR's. This is captured by the following metric weights: $\omega P = 0.99$, $\omega E = 0.01$. The payload is fixed at 48 Bytes but the message period is varied between 15 seconds and 32 minutes. The results are presented in Fig. 8a, 8b, and 8c using a logarithmic horizontal axis.

We can see in Fig. 8a that both ADR and AAPC's PDR performance is severely impacted as message period is decreased. However, while they achieve similar levels of performance down to 1 message per minute, ADR's performance drops off much more dramatically below this point, with AAPC achieving 2.7x the PDR.

The results in Fig. 8b help to explain why this is the case. As message period decreases, the average SNR at a given parameter set will decrease. ADR responds to this by increasing a node's signal strength, either by increasing TXP or SF, or both. Higher SFs consume more time-on-air, and higher TXPs increase the average noise, both of which amplify the problem and worsen congestion. This is reflected in ADR's increase in ECP at lower message periods. AAPC, which uses a different mechanism entirely, actually decreases its ECP.

Fig. 8c again demonstrates that AAPC requires more downlink messages than ADR in the given configuration. This particular plot has been scaled relative to message period to avoid the loss of detail at the lower end of the horizontal axis. As expected, ADR's downlink messaging drops off severely as fewer uplink messages are successfully delivered.

Although a message period of 15 seconds seems unrealistic for most applications, congestion can be achieved in other ways, including; increasing the number of nodes, deploying nodes further from the base station, or increasing the payload length.

4.7. Overhead control

AAPC utilises downlink messaging from the base station to the node as a part of the feedback approach. Keeping these messages to a minimum, and indeed below regulatory duty cycle limits, is necessary if it is to be feasibly implemented in practice. The preceding simulations revealed that, although AAPC's downlink messages consume very little airtime, it still has more overhead than ADR.

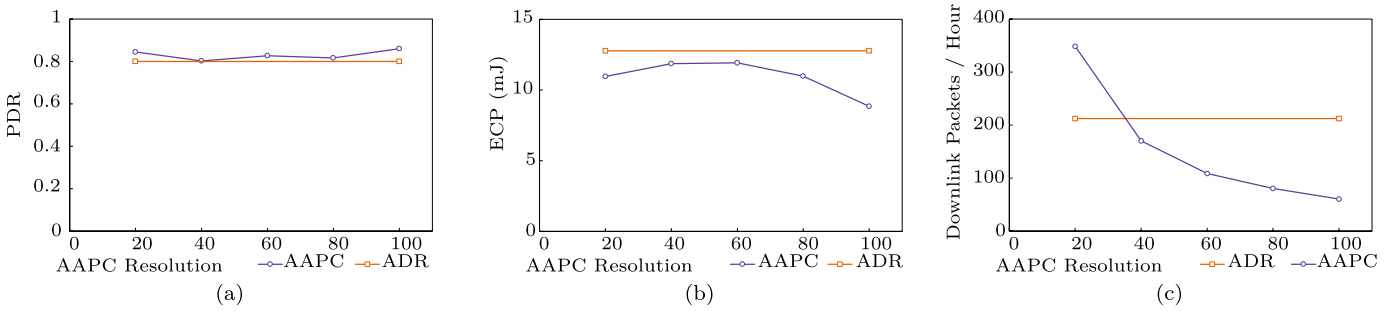


Fig. 9. (a) Mean network PDR against AAPC resolution; (b) Mean network ECP against AAPC resolution; (c) Scaled downlink packets against AAPC resolution.

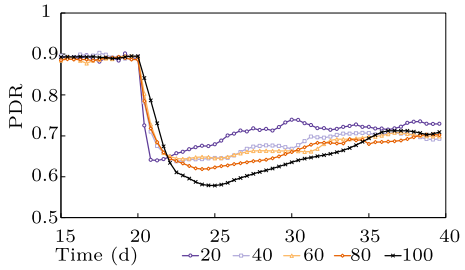


Fig. 10. Effect of resolution on responsiveness.

AAPC's PDR measurement resolution has been set to 20 samples to align it with ADR's *maxSNR* sample resolution. AAPC's measurement resolution is conceptually very similar to maxSNR, in that it represents the number of packet samples the base-station requires before an estimate of performance can be computed. In the case of ADR/maxSNR, this is the signal-to-noise ratio of the received packets. For AAPC, it is the packet delivery ratio. In terms of timing, with a message period of 10 minutes, a resolution of 20 corresponds with 200 minutes between PDR estimates.

PDR resolution does not necessarily need to be set to 20. Fig. 9a, 9b, and 9c present the results of a set of simulations which evaluate the downlink messaging overhead against different PDR resolutions. The message period is 10 minutes and the payload length is 16 Bytes, and all nodes use the following metric weights: $\omega P = 0.9$, $\omega E = 0.1$.

The PDR and ECP results in Fig. 9a and 9b respectively, show performance similar to that observed in previous simulations. Fig. 9c however, reveals a more interesting relationship. It demonstrates that, if the measurement resolution is increased, AAPC's downlink messaging, and hence overhead, is significantly reduced, below that of ADR.

Fig. 10 further explores the effect of AAPC's measurement resolution on responsiveness. The experimental scenario follows scenario 1, with significant noise introduced 20 days into the simulation. Each trace corresponds to a different measurement resolution between 20 and 100. Note that the axis limits have been selected to highlight the region of greatest interest. It can be observed that, with a resolution of 20, it takes AAPC 10 days after interference is introduced to settle on its preferred configuration. When measurement resolution is increased to 100, it takes 18 days. The other measurement resolutions sit somewhere in between. Armed with this information, together with the results from Fig. 9c, design decisions can be made which utilise the trade-off between responsiveness and overhead.

4.8. Implementation overhead

To evaluate the suitability of implementing the proposed approach on nodes, which are typically devices with low computational power, an experiment was carried out using physical hardware. The hardware contains an ARM Cortex M0+ based microcontroller, clocked at 32 MHz, with 20 kBytes of SRAM. This is relatively modest performance compared to currently available technology but is typical of low power radio transceiver + microcontroller modules. The complete process, starting from the base-station correction message, through to the computation of the best parameter set, has been implemented. The experiment demonstrated that this process takes 3.2 ms to complete. Compared to the time taken to send an 8 Byte packet upstream at the highest data rate (SF7) - 56 ms, confirms that the proposed approach takes an insignificant amount time to execute, and is suitable for the target platforms.

4.9. Evaluation summary

The results presented in this section demonstrate clear advantages of AAPC when compared to ADR. Even in the static phases of the simulations, AAPC demonstrates slight improvements in PDR, and notable improvements in ECP. However, the motivation behind AAPC, where it demonstrates its superiority, is in its ability to adapt to network dynamics, and its ability to address the specific performance goals of the application (application awareness).

Its adaptation to network dynamics is best demonstrated in scenario 1, where it achieves 1.9x the PDR performance of ADR, when impacted by significant and variable interference.

Its application awareness is demonstrated in scenario 2, where nodes easily switch their priority, when required, from ECP to PDR, resulting in a sharp increase in PDR performance. It is again demonstrated in scenario 3, where three different applications, with unique application requirements, co-exist on the same network, each achieving the metric focus it needs.

The congestion response evaluation highlights AAPC's advantage when the network becomes congested. Additionally, it reveals that AAPC responds to congestion by reducing SF and/or TXP (as indicated by the ECP plot), whereas ADR increases SF and/or TXP, further increasing congestion.

A common feature of the evaluation results is that AAPC's overhead consistently exceeded ADR's. The final set of simulations demonstrate that, by increasing AAPC's PDR resolution, the overhead can be reduced significantly, below that of ADR's, without impacting the PDR and ECP performance.

5. Conclusion

One of the key objectives of modern Internet of Things (IoT) technologies is to support large-scale, geographically distributed systems. Low Power Wide Area Networks (LPWANs) technologies, such as LoRaWAN, are designed for such purposes. Such networks very often consist of thousands of nodes for different purposes and with different application requirements, and individual nodes most likely experience different communication conditions due to their

surrounding environment. In this paper, an adaptive algorithm is proposed to dynamically decide network parameters based on a node's application requirements and changing conditions. The parameters are selected in a way to enhance the network performance and to ensure each individual node can satisfy its own application requirements. The proposed algorithm is evaluated with a few realistic scenarios, and compared with the existing Adaptive Data Rate approach implemented by LoRaWAN. It is demonstrated that the proposed algorithm can achieve targeted performance in both packet delivery ratio and energy consumption, while at the same time outperforming ADR. It is validated that the proposed approach is able to adapt to changing application requirements at runtime, and maintains network performance over time, while satisfying application requirements.

While comparison with ADR has been the most straightforward, there are other approaches which aim to surpass its performance. Due to differences in simulation parameters, a direct comparison cannot be made without implementing all approaches within the same simulation framework. This will be an avenue of future work. Next steps will also include qualifying the proposed approach on a large scale, physical network.

CRediT authorship contribution statement

Ameer Ivoghlian: Conceptualization, Formal analysis, Investigation, Methodology, Software, Writing – original draft. **Kevin I-Kai Wang:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Zoran Salcic:** Conceptualization, Supervision, Writing – review & editing.

Declaration of competing interest

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

References

[1] Simpy - discrete event simulation for python, https://simpy.readthedocs.io/. (Accessed 30 September 2010).

[2] K.Q. Abdelfadeel, V. Cionca, D. Pesch, Fair adaptive data rate allocation and power control in LoRaWAN, in: 19th Int. Symp. WoWMoM, IEEE, 2018, pp. 14–15.

[3] M.C. Bor, U. Roedig, T. Voigt, J.M. Alonso, Do LoRa low-power wide-area networks scale?, in: Proc. 19th Int. Conf. MSWiM, ACM, 2016, pp. 59–67.

[4] Semtech Corporation, LoRaWAN – simple rate adaptation recommended algorithm, https://www.thethingsnetwork.org/forum/uploads/default/original/2X/7/7480e044aa93a54a910dab8ef0adfb5f515d14a1.pdf, 2016.

[5] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, I. Tinnirello, Impact of LoRa imperfect orthogonality: analysis of link-level performance, IEEE Commun. Lett. 22 (4) (2018) 796–799, https://doi.org/10.1109/LCOMM.2018.2797057.

[6] F. Cuomo, M. Campo, A. Caponi, G. Bianchi, G. Rossini, P. Pisani, EXPLoRa: extending the performance of LoRa by suitable spreading factor allocations, in: 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2017, pp. 1–8.

[7] B. Dix-Matthews, R. Cardell-Oliver, C. Hübner, LoRa parameter choice for minimal energy usage, in: RealWSN, RealWSN'18, ACM, 2018, pp. 37–42.

[8] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, L. Hanzo, A survey of multi-objective optimization in wireless sensor networks: metrics, algorithms, and open problems, IEEE Commun. Surv. Tutor. 19 (1) (2017) 550–586, https://doi.org/10.1109/COMST.2016.2610578.

[9] V. Hauser, T. Hégr, Proposal of adaptive data rate algorithm for LoRaWAN-based infrastructure, in: 5th Int. Conf. FiCloud, IEEE, 2017, pp. 85–90.

[10] M. Iqbal, M. Naeem, A. Anpalagan, A. Ahmed, M. Azam, Wireless sensor network optimization: multi-objective paradigm, Sensors 15 (7) (2015) 17572–17620, https://doi.org/10.3390/s150717572, https://www.mdpi.com/1424-8220/15/7/17572.

[11] D.-Y. Kim, S. Kim, H. Hassan, J.H. Park, Adaptive data rate control in low power wide area networks for long range IoT services, J. Comput. Sci. 22 (2017) 171–178, https://doi.org/10.1016/j.jocs.2017.04.014, arXiv:1011.1669v3.

[12] L. Krupka, L. Vojtech, M. Neruda, The issue of LPWAN technology coexistence in IoT environment, in: 17th Int. Conf. Mechatronics, Prague, IEEE, 2016, pp. 1–8, http://ieeexplore.ieee.org/abstract/document/7827866/.

[13] S. Li, U. Raza, A. Khan, How agile is the adaptive data rate mechanism of LoRaWAN?, in: GLOBECOM, IEEE, 2018, pp. 206–212.

[14] A. Mahmood, E. Sisinni, L. Guntupalli, R. Rondón, S.A. Hassan, M. Gidlund, Scalability analysis of a LoRa network under imperfect orthogonality, IEEE Trans. Ind. Inform. 15 (3) (2019) 1425–1436, https://doi.org/10.1109/TII.2018.2864681.

[15] A.-I. Pop, U. Raza, P. Kulkarni, M. Sooriyabandara, Does bidirectional traffic do more harm than good in LoRaWAN based LPWA networks?, in: GLOBECOM, IEEE, 2017, pp. 1–6.

[16] U. Raza, P. Kulkarni, M. Sooriyabandara, Low power wide area networks: an overview, IEEE Commun. Surv. Tutor. 19 (2) (2017) 855–873, https://doi.org/10.1109/COMST.2017.2652320.

[17] B. Reynders, W. Meert, S. Pollin, Power and spreading factor control in low power wide area networks, in: 2017 IEEE International Conference on Communications (ICC), ISSN 1938-1883, 2017, pp. 1–6.

[18] E. Sallum, N. Pereira, M. Alves, M. Santos, Improving quality-of-service in LoRa low-power wide-area networks through optimized radio resource management, J. Sens. Actuator Netw. 9 (1) (2020) 10, https://doi.org/10.3390/jsan9010010.

[19] Semtech Corporation, SX1272/3/6/7/8: LoRa modem designer's guide, https://lora-developers.semtech.com/library/product-documents/, 2013.

[20] Semtech Corporation, LoRa modulation basics, https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1276, 2015.

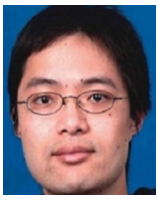
[21] Semtech Corporation, SX1276-7-8-9 datasheet, https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1276, 2019.

[22] M. Slabicki, G. Premsankar, M. Di Francesco, Adaptive configuration of lora networks for dense IoT deployments, in: NOMS, IEEE, 2018, pp. 1–9.

[23] A. Waret, M. Kaneko, A. Guitton, N. El Rachkidy, LoRa throughput analysis with imperfect spreading factor orthogonality, IEEE Wireless Commun. Lett. 8 (2) (2019) 408–411, https://doi.org/10.1109/LWC.2018.2873705.

[24] S. Wu, K.I.-K. Wang, A. Ivoghlian, A. Austin, Z. Salcic, X. Zhou, LWS: a LoRaWAN wireless underground sensor network simulator for agriculture applications, in: Proc. 16th Int. Conf. Ubiquitous Intelligence and Computing, IEEE, 2019, pp. 475–482.

Ameer Ivoghlian received his Bachelor of Engineering (Hons) and Master of Engineering degrees in Computer Systems Engineering at the University of Auckland, where he is currently a doctoral candidate. Prior to commencing his doctoral studies, Ameer worked as an electronics design engineer in the payments industry. His research interests include low power systems, machine learning, and the Internet of Things. His current research focus is on automatic configuration and management for large-scale wireless sensor networks.



Kevin I-Kai Wang received the Bachelor of Engineering (Hons.) degree in Computer Systems Engineering and PhD degree in Electrical and Electronics Engineering from the Department of Electrical and Computer Engineering, the University of Auckland, New Zealand, in 2004 and 2009 respectively. He is currently a Senior Lecturer in the Department of Electrical, Computer and Software Engineering, the University of Auckland. He worked in industries as a research engineer designing commercial home automation systems and traffic sensing systems from 2009 to 2011. His current research interests include wireless sensor network based ambient intelligence, pervasive healthcare systems, human activity recognition, behaviour data analytics and bio-cybernetic systems.

Zoran Salcic (Life Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electrical and computer engineering from Sarajevo University in 1972, 1974, and 1976, respectively. He is a Professor and the Chair of computer systems engineering with The University of Auckland, New Zealand. He has published more than 400 peer-reviewed journal and conference papers, and several books. His main research interests include various aspects of cyber-physical systems that include complex digital systems design, custom-computing machines, design automation tools, hardware–software co-design, formal models of

computation, sensor networks and Internet of Things, and languages for concurrent and distributed systems and their applications such as industrial automation, intelligent buildings and environments, and collaborative systems with service robotics and many more. He is a Fellow of the Royal Society of New Zealand. He was a recipient of the Alexander von Humboldt Research Award in 2010.