TECHNISCHE UNIVERSITÄT
CHEMNITZ

On Enforcing Reliability in Unidirectional WSNs: A MAC-Based Approach

Dissertation
zur Erlangung des akademischen Grades

Dr.-Ing.

Herr Dipl.-Ing. Philip Parsch
geboren am 19.September 1987 in Freiburg im Breisgau

Fakultät für Informatik
an der Technischen Universität Chemnitz

Gutachter:

1. Prof. Dr. Alejandro Masrur
2. Prof. Dr. Wolfram Hardt
3. Prof. Henry Hoffmann, PhD

Tag der Verteidigung:

28.05.2019

URL: http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa2-341799

Acknowledgments

First of all, I would like to thank my advisor Prof. Masrur for the continuous support of my doctoral thesis and the associated research, for his patience, encouragement and immense knowledge. His guidance has helped me throughout my research and writing this work. I could not have imagined a better advisor and mentor for my doctoral thesis.

I am also sincerely thankful for Prof. Hardt for helping me in many ways and grant me incredible opportunities, starting with co-financing with my job, providing valuable tips for my work, organizing numerous seminars and workshops at which I could present my work and meet new people. All these things have helped me a lot and I couldn't be more grateful. Thank you very much.

I would also like to thank my colleagues for all the good times and laughter we have shared that helped me keep being sane. You have made me aware that I am not alone in this, but also others must survive the "battle" of the doctorate. A sorrow shared is a sorrow halved.

And lastly, I would like to thank my family and my girlfriend for emotionally supporting and helping me get through the difficult times of my doctorate. It is good to have an open ear for problems and to always get the right answers. You saved me from becoming unbalanced even in the most stressful times. Thank you very much.

Abstract

With the advent of Internet of Things (IoT), an increasing number of devices start exchanging information. This puts emphasis on wireless sensor networks (WSNs) to facilitate the interaction with the environment in varied application scenarios such as, for example, building and home automation among others. In this context, a reliable communication is usually required, i.e., it is necessary to guarantee that packets arrive within a specified maximum delay or deadline. In addition, since nodes are usually battery-powered and deployed in large numbers, they must be cost-effective and economize on energy, which requires nodes to have a low complexity.

In this context, unidirectional communication, i.e., where nodes either send or receive data, seems to be an interesting solution. Since no elaborate feedback mechanisms such as carrier sensing, acknowledgments and retransmissions schemes are possible, complexity, costs, energy consumption and communication overhead are reduced in a considerable manner. On the other hand, however, packet loss becomes more likely making such networks strongly unreliable. To overcome this predicament, two MAC (Medium Access Control) protocols are proposed, namely DEEP and RARE. These consist in nodes transmitting their data as sequences of redundant packets with carefully selected inter-packet separations leading to robust transmission patterns that enable reliable communication. In the case of DEEP, full (100 %) reliability can be guaranteed, i.e., there is no data loss, which is particularly useful for safety critical applications. RARE, on the other hand, is designed for applications that tolerate some amount of data loss and can be configured to a reliability $< 100\,\%$, i.e., to a certain probability on successful data delivery. This allows improving other aspects of the network, such as energy consumption, communication delays, etc.

In contrast to solutions from the literature, the proposed protocols do not pursue a best-effort approach, but rather provide an analytical framework to assess the performance (i.e., reliability, energy consumption, etc.) of the network. In addition, the proposed protocols are based on more general models that allow describing arbitrary node types with different deadlines and packet lengths leading to a provable higher performance. These and other benefits are illustrated by the means of extensive numerical experiments and simulations based on the OMNeT++ framework.

List of Acronyms

IoT	Internet of Things
WSN	Wireless Sensor Network
QoS	Quality of Service
MAC	Medium Access Control
PHY	Physical layer (in the OSI model)
OSI	Open Systems Interconnection model
DEEP	Deterministic Protocol
RARE	Random Reliable Protocol
bi-DEEP	Bidirectional Deterministic Protocol
bi-RARE	Bidirectional Random Reliable Protocol
CSMA	Carrier-Sense Multiple Access
TDMA	Time-Division Multiple Access
eMAC	Extensive MAC
API	Application Programming Interface
WBAN	Wireless Body Area Network
RFID	Radio-Frequency Identification
FSK	Frequency-Shift Keying
DSSS	Direct-Sequence Spread Spectrum

Contents

Chapter 1.

Introduction

Since the first controlled transmission of radio waves by Heinrich Hertz in 1886, wireless communication has quickly developed into an indispensable technology for exchanging information between electronic devices. This is mainly due to its distinctive features: high flexibility and mobility. Wireless devices can be easily integrated into existing infrastructure, which, in contrast to classic wired connections, does not require costly cable installation. In addition, devices are not tied to a fixed position, but can move freely within a certain range from each other. This enables a series of applications and technologies that would otherwise not be possible, such as mobile telephony and internet (e.g., 4G and 5G), contactless payment, Global Positioning System (GPS), and many more.

Current trends further promote spreading wireless communication. In particular, the number of devices in our environment is constantly increasing together with the amount of information they exchange. Starting with smartphones and computers that communicate via WiFi, today many other devices begin sharing information: door access and security systems, fire detectors, vehicles and road infrastructure and many more. This trend is expected to continue until all electric devices are interconnected accomplishing the vision of Internet of Things (IoT) [15]. According to market research, the number of connected IoT devices is predicted to reach 75 billion by 2025 [54] and 125 billion by 2030 [40], and will therefore have a major impact on our lives in the future.

There are many different applications and technologies based on wireless communication. This paper focuses on wireless sensor networks (WSNs), a well-established field of research and a common type of network used to retrieve information from the environment. More precisely, WSNs consist of several sensor nodes that monitor the environment and report data back to a central location for processing. This creates a feedback service that facilitates the interaction with the environment, making WSNs not only a popular choice as standalone networks, but these are also often used as a part of other technologies and applications, such as IoT.

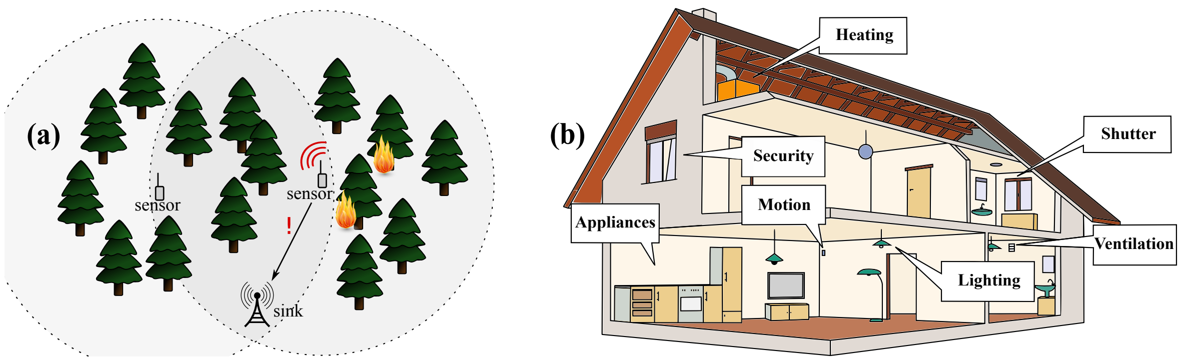


Figure 1.1.: Two exemplary applications for WSNs: (a) forest fire detection [50] and (b) home and building automation [66].

WSNs come in all sizes, shapes and are being used in many different applications. In general, they are well suited for monitoring tasks, for example, controlling air quality, moisture, temperatures, etc. [39], or supervising traffic [55] and parking [75]. Fig. 1.1 shows two further examples: (a) forest fire detection and (b) home and building automation. In (a), WSNs can detect bush and forest fires using gas/temperature sensors so that they can be extinguished more quickly before they spread [50]. In (b), electrical appliances within a building are controlled, improving heating, ventilation, security and enabling a more elaborate control [66]. For example, lights can be automatically turned on at night when a user enters a room. Similarly, appliances, e.g., TV, radio, coffee machine, etc., can be switched off when a user leaves home, etc. In summary, WSNs have a wide spectrum of possible applications and are used for many other technologies such as IoT, i.e., wherever a connection to the environment must be established. For these reasons, WSNs have become an important field of research over the years, which is still highly active today.

Using wireless communication has many advantages such as the previously mentioned mobility and flexibility. Nevertheless, there are also several disadvantages, in particular, reliability is lower compared to wired connections. This is mainly due to the open transmission medium (air) which must be shared between devices. If it happens that multiple transmissions take place simultaneously on the same frequency — devices are typically not aware of each other — data is most likely corrupted and therefore lost. Considering that this effect is highly stochastic, detecting and repairing faulty transmissions is a challenging task requiring special attention during network design and operation.

In oder to better understand these challenges, let us have a look at the typical requirements of WSNs as depicted in Fig. 1.2:

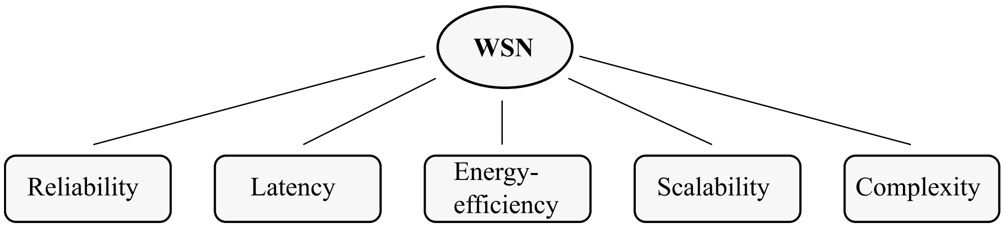


Figure 1.2.: The 5 basic requirements of WSNs.

In general, there exists a large number of different requirements caused by the sheer amount of possible applications together with several environmental factors, such network location (e.g., indoor vs. outdoor), life time (e.g., several days vs. years), etc. However, these requirements can be grouped into the following five *main* categories:

- **Reliability** describes the probability that data is successfully delivered within a specified time bound or deadline. This metric combines data loss, i.e., the chance of losing a packet due to simultaneous transmissions or interference, with the age of data, i.e., whether it is conveyed on time before being outdated.
- **Latency** describes the time required for exchanging data or vice versa, how much data can be transferred in a given amount of time. For example, a video transmission requires a lower latency (and higher speed) than a thermometer reporting a temperature.
- **Energy Efficiency** describes how well the network can economize on energy. Since nodes are usually battery-powered and maintenance can be difficult and expensive, this value defines a node's life-time and its operational costs.
- **Scalability** describes the performance impact when changing the network size (i.e., the number of nodes). This metric is especially important in mobile systems, where nodes may enter or leave the network dynamically.
- **Complexity** determines the costs of implementing and operating a network. Low complexity is desirable as it allows using cheaper and less powerful hardware, etc.

There are other, more application-specific requirements, such as security, i.e., how well the network is protected from unwanted listeners or attackers, or robustness, i.e., how well operation can continue when some nodes fail (e.g., run out of battery). In general, these strongly depend on the use case, e.g., a military-operated network typically requires a higher level of security than a private weather station, which, however, must have lower costs. Unfortunately, many of these requirements are contradictory meaning that increasing one attribute will worsen others. For example, implementing a higher transmission speed typically requires more energy and can increase packet loss, which lowers reliability. As a consequence, it is not possible to maximize all attributes of a network, but instead performance is a compromise based on how important certain aspects are for a specific setting or application.

Now, there are several different ways to meet requirements in a WSN. For example, the physical layer (PHY) can be changed to a more robust modulation scheme in harsh environments or applications can be programmed to tolerate data loss to some extent. In this work, the focus is on medium access control (MAC), i.e., the second layer in the OSI model [5]. More specifically, this defines a policy of when and how long nodes transmit/receive, what data they send and how they behave when failures occurs, e.g., retransmit, wait, etc. Compared to the PHY layer, which employs modulation and encoding that is often highly hardware-dependent, the MAC layer is typically hardware-independent which allows using available off-the-shelf hardware without modifications. This facilitates design, testing and integration into existing systems, since typically only software must be changed, which is easier than modifying hardware. In addition, since the MAC is still a very low layer in the OSI communication model, it generally allows for a larger impact on a system's performance than higher levels such as the application layer.[1] This results in a lot of potential for possible improvements.

1.1. Motivation

This work aims to improve the quality of service (QoS) of WSNs and, in particular, enable reliable communication at low complexity between nodes in a network. To this end, two different MAC protocols were developed in the course of this work, each for a particular set of applications with specific requirements.

Now, the question is why to design another set of MAC protocols, when there are already so many of them. In particular, since wireless MAC protocols are an active research area for many years now — modern research started at around 1980 [26], i.e., more than 30 years ago — there exists a plethora of different solutions for a wide variety of applications. While some of these solutions are very specialized and only suitable for particular settings, others have become very popular and are widely-used nowadays. For example, TDMA (time division multiple access) and CSMA (carrier sense multiple access) are very prominent protocols being used in most wireless applications today. Defined and regulated by multiple standards, such as 802.11 (WiFi, CSMA) or 802.16a (WiMax, TDMA), these can be found in toys and computer networks (CSMA) or mobile telephony and internet (TDMA) among others.

So why does it make sense to implement yet another set of MAC protocols? The answer is that although there are so many protocols, there is no perfect solution for all applications. As we have discussed before, this is because each application has different requirements, which are often contradictory, e.g., speed vs. energy-efficiency. It is consequently not possible to create one universal MAC, but there have to be multiple solutions, as there will always be different types of problems to be solved. In addition, many existing solutions make restrictive assumptions or have major drawbacks that limit performance or result in certain properties

[1] Note that the higher the layer of communication is, the smaller the achievable increase in performance. That is, since each layer builds on top of another (lower) layer, it is limited to that lower layer's performance in the worst case.

not being offered. It therefore still makes sense to introduce new protocols to not only improve certain quality aspects of existing solutions, but also to open up new areas of application.

In the following, we will further discuss some of the major drawbacks that many existing solutions have, focusing in particular on those that the proposed protocols of this work aim to solve or improve. One of these drawbacks is the lack of adaptability, i.e., not being able to adapt to varying environmental factors, etc. In general, different networks are never exactly the same, but these are typically deployed at different locations with varying conditions or have different number of nodes, etc. Using the same MAC parameters without modifications will most likely lead to sub-optimal performance, which, in some cases, result in up to 70 % deterioration [33] [57]. Especially general-purpose protocols such as CSMA, which have been designed to be suitable for a wide range of applications, need to be configured properly for good performance [9]. However, many protocols cannot be adapted or allow for only small changes limiting possible improvements.

Another disadvantage of many protocols is that these pursue a best-effort approach, i.e., they present methods to increase performance, but are not able to specify or quantify it. For example, in [31] nodes are equipped with multiple antennas or in [93] sink nodes are placed at selected locations in order to increase reception quality and, therefore, improve reliability. However, instead of providing a mathematical analysis allowing to calculate the expected system behavior in advance, these approaches evaluate performance in simulation. This requires the network to be already set up, which might not be feasible in some cases. In particular, during network design, it is advantageous to assess performance before an actual deployment to save costs and speed up the development process.

In addition, many protocols are very complex, e.g., they require synchronization or control messages to be sent. For example, with TDMA, nodes share a common clock by periodically synchronizing to beacon messages sent by the sink. This allows to effectively coordinate transmissions and prevent collisions — each node has a separate slot to transmit. However, this also comes at the expense of higher complexity, which complicates implementation and increases costs. Now, in order to reduce complexity, unidirectional communication seems to be an interesting solution. In such networks, nodes can only send *or* receive, i.e., data transfers are one-way only from transmit-only sensors to receive-only sinks. As a result, no feedback mechanisms can be implemented such as carrier sensing or acknowledgments. This reduces complexity and costs considerably and has the advantage of improving energy efficiency, since high protocol overhead is avoided and nodes do not need to monitor the communication channel [14] [51] [95]. However, on the other hand, transmissions cannot be coordinated resulting in higher packet loss and decreased reliability. As a result, most existing protocols are either not reliable or difficult to configure properly [10].

There are many other disadvantages that could be listed here. In summary, it can be said that existing solutions can still be improved in many respects, e.g., to compensate for the disadvantages mentioned above. This work addresses some of these problems and introduces two MAC protocols that offer a novel combination of features improving existing solutions in several ways. Details are discussed in the next section.

1.2. Contributions

This work presents two MAC protocols that enable reliable communication in low-cost, unidirectional networks, i.e., an application area where other protocols typically fail to provide any QoS guarantees. More specifically, these allow for performance improvements compared to existing solutions from the literature, and provide a novel combination of features that were not available before. In the following, we will first discuss the principles of the proposed protocols, list their unique properties and features and, finally, conclude with the remaining contributions.

The presented protocols are designed to enable reliable communication at very low complexity to be suitable for low-cost networks. To this end, they are based on unidirectional communication, where nodes forgo any kind of feedback such as carrier sensing or acknowledgment. Now, in order to prevent possible reliability problems — existing reliable approaches such as TDMA cannot be used — two special MAC protocols are presented in this work:

- **DEEP** (deterministic protocol) is designed for full reliability, i.e., it can guarantee that data always reaches its destination within a specified deadline (when neglecting external interference). To this end, each node transmits sequences of redundant packets with constant inter-packet (i.e., backoff) times, thus generating a unique, collision-free transmission pattern.

- **RARE** (random reliable protocol) can be configured to a user-specified reliability of less than 100 %, i.e., it is designed for applications that tolerate some data loss, which, in return, allows improving other performance metrics such as energy efficiency. Similar to DEEP, nodes transmit sequences of redundant packets, however, inter-packet times are random and packet numbers typically smaller.

Both protocols are designed for simplicity and low costs, as commonly required by commercial products such as home automation. In the case of DEEP, full reliability can be guaranteed under the assumption that external interference can be neglected — such networks are commonly located indoors and, therefore, well-shielded by walls. This is particularly useful for safety-critical applications, for example, heating control or security in a home automation network. On the other hand, if devices do not need full reliability, e.g., they can tolerate some data loss, packet numbers can be reduced to save energy in return.

The RARE protocol is also designed to enable reliable data transport in low-cost, unidirectional networks. However, in contrast to DEEP, it makes use of the fact that many application do not require full reliability, but can tolerate some data loss. For example, sensors transmitting heart rate and running speed, etc. to a health monitoring application or temperature sensors reporting data to a weather station. Losing some packets in this context does not negatively impact the system's performance, as long as a minimum reliability is not undershot. This avoids the high overhead of full reliability, i.e., the extra time and energy that must be invested, and allows using a (mathematically) less complex probabilistic approach.

Note that the focus of this work is on unidirectional networks, since these have been proven to significantly reduce complexity, energy-consumption and costs [14] [51] [95], which are all objectives of this work. However, it was observed that the principles of DEEP and RARE can also be effectively used in bidirectional networks. For this reason, Appendix B contains an extension of DEEP and RARE for bi-directional networks, called bi-DEEP and bi-RARE, which implement carrier sensing and acknowledgments. A short comparison between all protocols is later performed in Chapter 6.

As mentioned before, the proposed protocols DEEP and RARE offer a unique combination of features, which many other state-of-the-art approaches cannot provide. These are:

- **Reliability:** Both protocols are reliable, i.e., they can guarantee either full or a configurable level of reliability in the worst case.

- **Low Complexity:** By omitting synchronization and other control overhead, the presented MACs are easy to implement and suitable for low-cost networks.

- **Analytic Framework:** The proposed protocols are based on mathematical frameworks, which allow calculating the expected system behavior in advance, i.e., without requiring an actual test bed. In addition, optimal network parameters can be determined to adapt the system to a given environment for improved performance. This includes practical factors, such as external interference and clock drift.

- **Energy Efficiency:** Packet numbers can be adjusted to lower energy consumption (at the cost of reliability) when needed, e.g., when slight data loss can be tolerated.
- **Flexibility:** In contrast to many protocols from the literature, the proposed MACs can account for arbitrary node types with varying packet lengths and deadlines allowing for more accurate network models and improved performance.

The remaining contributions can be summarized as follows. Each presented MAC protocol is evaluated analytically and by simulation using the OMNeT++ framework [85]. This includes a comparison to existing approaches from the literature, such as the commonly used CSMA and TDMA schemes. Results show that the protocols of this work considerable improve reliability, energy-efficiency, etc. and, therefore, typically allow for a higher QoS in WSNs compared to other state-of-the-art approaches.

1.3. Models and Assumptions

This section describes the network models and underlying assumptions of DEEP and RARE. To this end, all parameters common to both MAC protocols are introduced, while all other MAC-specific values are explained later in the corresponding theory chapters, i.e., in Chapter 4 for DEEP and Chapter 5 for RARE. At the end of this section, a model for external interference is presented.

The underlying WSN consists of n transmit-only nodes and multiple receive-only sinks that are connected in a single-hop (star-topology) fashion, as shown in Fig. 1.3. Data transfers can either be triggered by events or periodically depending on the application. To this end, each sink constantly monitors the communication channel to be able to receive data packets directly without requiring special mechanisms, such as wakeup messages. In many applications, this assumptions does not pose any additional restrictions, since sinks are usually equipped with a larger battery or connected to a continuous power supply. For example, in home automation, sinks are typically connected to a house's electric network.

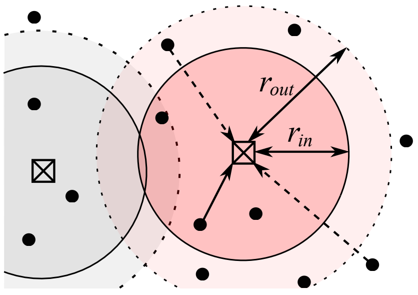


Figure 1.3.: Example of a single-hop WSN with transmit-only nodes (solid circles) and sink nodes (checked boxes): r_{in} represents the range within which a sink collects packets, while r_{out} indicates the range in which transmitters can interfere with each other, e.g., at simultaneous transmissions.

To ensure normal (error-free) operation, data must arrive at the corresponding sink within an upper time bound or deadline d with $d \in \mathbb{R}_{>0}$. If this time is exceeded, the package content is considered to be outdated and must be discarded. This decreases the QoS of the system — it will be less responsive — and might cause errors, since too much time has passed between updates. For example, in the intelligent assembly line from Section 6.1, where the movement of mobile robots is controlled wirelessly, missing updates for a longer period of time can lead to collisions between robots and workers. In the case of RARE, at least one packet must arrive at the sink with a certain probability p in the worst case. For DEEP, which guarantees full reliability, this probability is $p = 1$ (100 %) and at least one packet always arrives in the worst case.

Since individual packets can be corrupted, for example, by simultaneous transmissions from other nodes, each node i transmits a sequence of k_i redundant[2] packets within its deadline d_i with $1 \leq i \leq n$, $1 \leq k \leq n$ and $k_i \in \mathbb{N}$. This increases the chance that at least one is received correctly and, hence, improves reliability. The time between transmissions, called inter-packet times t_i, must be selected accordingly to avoid too many collisions between retransmissions; here, $t_i \in \mathbb{R}_{>0}$ and $t_i \leq d_i$ hold. To this end, DEEP and RARE present different methods to calculate these values, however, more on this later in Chapter 4 and 5.

Transmitting a packet takes a certain amount of time, depending on the transmission speed and the size of the data. This time, called packet length, is denoted by l_i with $l_i \in \mathbb{R}_{>0}$ and can differ for each node i in the network. Note that $l_i \leq t_i$ must hold for any i, i.e., each transmit-only node must be able to send its full packet within a time interval equal to its inter-packet separation.

Packets can lost as a consequence of interference on the communication channel. This can either originate from within the network, i.e., when other nodes with overlapping space and frequency ranges (see r_o in Fig. 1.3) transmit simultaneously, or from outside the network, e.g., from neighboring systems. To this end, similar to many other approaches from the literature [21] [94], it is (pessimistically) assumed that packets cannot be recovered in the case of collisions, i.e., there is no capture effect, since this is highly stochastic, depending on signal strength, node placement and on which parts of the packets overlap [27] [31]. For this reason, capture effect leads to erratic and non-reproducible results, making it impossible to provide any worst-case guarantees as aimed in this work.

Lastly, it is assumed that nodes are activated only once within a sufficient large time interval $t_{act} \geq d_{max}$ with d_{max} being the largest deadline of all nodes in the network, i.e., $d_{max} = \max\limits_{i=1}^{n}\{d_i\}$. This is a logical assumption since multiple activations of the transmitters lead to unnecessary interference. Of course, t_{act} should not lead to unacceptable delay and should be tolerable by the application. However, more on this in the individual theory chapters.

Parameter	Description
n	number of transmit-only nodes in the network
p	reliability ($p < 1$ for RARE and $p = 1$ for DEEP)
k	number of packets per sequence
d	deadline by which at least one packet must arrive at the sink
l	packet length, i.e., time needed for transmitting a number of bytes
t	inter-packet time, i.e., time between two packet transmissions
t_{act}	time after which a node can be triggered anew

Table 1.1.: MAC parameters common to both DEEP and RARE.

Table 1.1 summarizes the previously mentioned parameters that are common to both protocols. Again, all other MAC-specific parameters will be discussed later in the individual theory chapters, i.e., in Chapter 4 for DEEP and Chapter 5 for RARE.

The last part of this section presents a model of external interference that will later be used in the theory part of DEEP and RARE. To this end, let us first start with a short definition: External interference is defined as noise on the communication channel that can

[2]Packets must not necessarily be redundant, but can also contain updated data. This can be regarded as oversampling with an increase in quality, if all packets of the sequence are received, and minimum acceptable functionality, if only one packet is received within a deadline d.

occur, for example, when microwaves, wireless toys, etc. are turned on, or when there exist neighboring WSNs that have not been regarded during the design phase. This noise can alter and corrupt data transmissions, resulting in a deteriorated system performance. Although there are numerous ways to mitigate such effects, e.g., by selecting a more robust modulation scheme or a less noisy frequency channel, as discussed in Section 2.2.2, these cannot always be avoided. For this reason, a model of external interference is presented that is later used by both DEEP and RARE to adapt their parameters and, hence, increase their robustness to external interference.

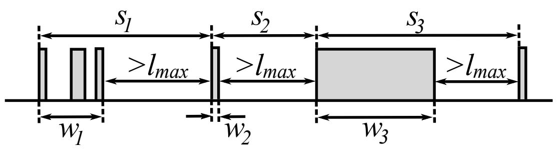


Figure 1.4.: An exemplary sequence of external interference pulses at the communication channel. In this case, the maximum possible duty cycle σ is equal to $\frac{w_3}{s_3}$. Note that external interference pulses that are separated by less than l_{max} are considered to be one single large pulse, since no packet can be sent in this short time interval. This is the case of the first three pulses in the example, which are merged into on single pulse of width w_1.

Now, in order to model external interference, its maximum *duty cycle* — denoted by σ — must be determined, i.e., the greatest possible ratio between pulse width w_i to inter-pulse separation s_i of external interference — see Fig. 1.4:

$$\sigma = \max_{\forall i} \left(\frac{w_i}{s_i} \right), \tag{1.3.1}$$

where $i \in \mathbb{N}_{>0}$ is an index identifying the particular pulse. This σ can be obtained, for example, by measuring at the communication channel for a sufficiently large time window; or this may also be known from previous experience. Note that external interference pulses separated by less than l_{max} (i.e., the longest packet of all nodes) are considered to be one single large pulse. This is because minimum overlapping with an external interference pulse yields packet loss — again, no capture effects are regarded. As a result, no data packet can be sent between such pulses as shown in Fig. 1.4.

Note that there are other models of external interference in the literature that can be used for DEEP and RARE as well — an overview can be found in [20]. For example, [22] presents a method to accurately classify and predict interference and [42] proposes a model that incorporates effects from the physical layer for higher accuracy. However, [22] has the disadvantaged of requiring continuous carrier sensing, which strongly increases energy consumption and complexity, and [42] requires knowledge about the physical layer, which might not always be available. In contrast, this work offers a simplified model with reduced complexity that provides a worst-case probability of encountering interference on the communication channel (i.e., σ). This facilitates integration into DEEP and RARE, since these are also based on a worst-case analysis.

1.4. Structure of the work

The rest of this work is structured as follows. Chapter 2 covers the basic fundamentals required for understanding the theory of this work. To this end, it examines the challenges of wireless communication more closely, redefines the tasks of the MAC layer and discusses its possibilities and limitations. This also includes a discussion about the differences between uni-

and bidirectional communication. At the end of this chapter, the commonly-used protocols TDMA and CSMA are introduced, as these will later be used in simulation in Chapter 6.

Chapter 3 is concerned with related work, for which it first provides an overview of different working principles and classifications schemes of MAC protocols. The second part then covers commonly-used strategies and a selection of protocols similar to those presented in this work, including a discussion about their advantages, disadvantages and differences.

Chapter 4 and Chapter 5 present the unidirectional MAC protocols DEEP and RARE. Each protocol is introduced with a short overview followed by a stepwise derivation of their underlying models. To facilitate understanding, base models are derived first that do not consider arbitrary node types with different deadlines, packet lengths, etc., and neglect practical factors such as external interference. These are then extended separately in later sections. At the end of each chapter, the protocols and their key findings are summarized shortly.

Note that Appendix B introduces the protocols bi-DEEP and bi-RARE, which are bidirectional extensions of DEEP and RARE that use carrier sensing and acknowledgment schemes. Towards this, the first part briefly discusses the effects of carrier sensing and acknowledgments, followed by a theoretical part in which the underlying models are gradually derived. Finally, each protocol is briefly summarized at the end.

Chapter 6 evaluates DEEP and RARE and compares them to other protocols from the literature (including bi-DEEP and bi-RARE). First, an exemplary case study is introduced to demonstrate the design of a WSNs and obtain suitable values for the following experiments. Then, the compared protocols are discussed and two different types of experiments are performed: a numerical evaluation that analyses the worst-case behavior of DEEP and RARE and a simulation-based evaluation that analyses their average performance and compares them to other protocols from the literature. At the end, results are summarized.

Chapter 7 finally concludes this work by summarizing DEEP and RARE, discussing their pros and cons, and listing possible application areas. At the end of this chapter, an outlook to possible future extensions is provided.

Chapter 2.

Fundamentals

This chapter covers the basics of wireless communication that are important for understanding the rest of this work. To this end, the first section starts with an explanation of the communication flow in WSNs using the OSI network model. Next comes a discussion about the problems and challenges in wireless communication and the possibilities (and limitations) of MAC protocols to address these. Lastly, the differences between uni- and bidirectional communication are highlighted and the widely-use CSMA and TDMA protocols are introduced.

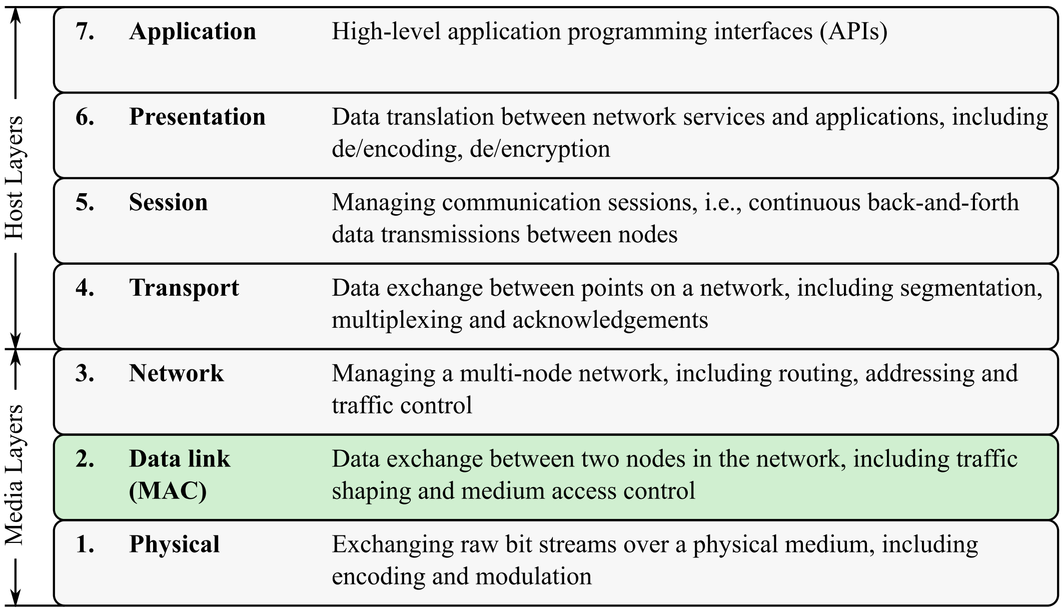
2.1. The OSI layer model

This section describes the communication flow in more detail, i.e., the different processing steps that information must undergo to be transmitted/received wirelessly. To this end, the Open System Interconnection (OSI) model [5] is used to conceptually describe the different processes within a node. As shown in Fig. 2.1, the OSI model consists of 7 individual layers or levels, each representing a specific set of functions and processes. The bottommost level, i.e., the physical layer, forms a connection to the hardware of the node, e.g., the transceiver circuitry, whereas the uppermost level connects to the application running on the node. All intermediate levels, either abstract the information when flowing upwards in the model, for example, convert a raw bit stream into packets and finally into dynamic data containers, or revert this abstraction when flowing downwards. Regarding wireless communication, transmitting data resembles a flow downwards, i.e., the data of the application has to be converted into a bit stream to be transmitted on the wireless channel, whereas receiving represents a flow upwards, i.e., the bit stream has to be demodulated and converted into information that can be read by the application.

In the following sections, the different OSI layers are examined in more closely, whereby the focus is on the MAC related layers 1-3. Note that a MAC protocol is typically represented by layer 2, however, some protocols also use mechanisms from other levels, for example, modulation [51] (layer 1), routing [24] (layer 3) or other cross-layer functions [8] (layer 1 to 3). The two protocols proposed in this work do not include such cross-layer functions, but these are solely based on layer 2. More on this later in Section 2.3.

2.1.1. Physical layer

The physical layer (PHY) is the lowest layer in the OSI model and defines the electrical and physical specification of the data connection. It is connected to the hardware of a node, i.e., the radio transceiver, where it controls modulation and encoding of data, which at this point is just a raw bit stream. The main task of the PHY is to efficiently transmit and receive such bit streams, which is a challenging task due to numerous physical effects that can possibly impair communication [84]. One such negative effect is multipath propagation where the receiver detects multiple versions of the transmitted signal that were reflected off walls or obstacles. Since these reflections are altered in frequency, phase, etc., they might overlap and corrupt the actual signal. Another major problem is noise, i.e., when a transmission is superimposed with random, unknown signals. This is a very common phenomenon, since


Host Layers
Media Layers

7.	**Application**	High-level application programming interfaces (APIs)
6.	**Presentation**	Data translation between network services and applications, including de/encoding, de/encryption
5.	**Session**	Managing communication sessions, i.e., continuous back-and-forth data transmissions between nodes
4.	**Transport**	Data exchange between points on a network, including segmentation, multiplexing and acknowledgements
3.	**Network**	Managing a multi-node network, including routing, addressing and traffic control
2.	**Data link (MAC)**	Data exchange between two nodes in the network, including traffic shaping and medium access control
1.	**Physical**	Exchanging raw bit streams over a physical medium, including encoding and modulation

Figure 2.1.: The 7-layer OSI model [5]. A MAC protocol is typically represented by level 2, however, there is no clear separation and some protocols include methods from layer 1 and 3, e.g., [8] and [51].

the wireless medium is shared by many nodes and even non-wireless devices can generate electromagnetic emissions, e.g., sparks in a rotating electric motor. In particular, if the received signal is very weak, which typically happens when nodes are located far apart, it can easily be distorted by interference. If this distortion is too strong, the receiver cannot decode the incoming data stream anymore and bit errors occur, which, in the worst case, lead to a failed reception. For more information about such effects and other problems in wireless communication, see Section 2.2.

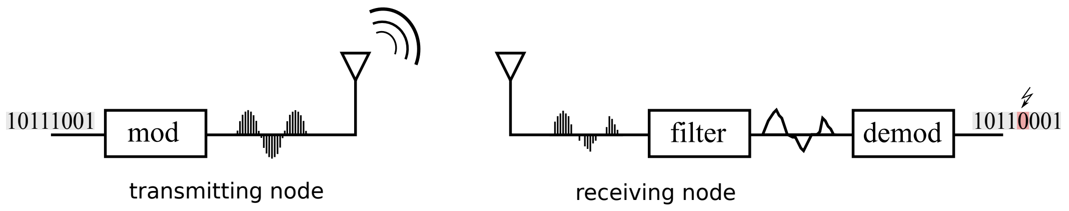


Figure 2.2.: Simplified illustration of the modulation (mod) and demodulation (demod) processes of a data transfer between two nodes. In this example, the wireless signal has been altered during transmission, e.g., by interference, which has caused a bit error in the received data stream.

Fig. 2.2 depicts the data shaping process of the PHY layer in more detail. Whenever data needs to be conveyed, the PHY of the transmitting node will receive a binary stream of information. Since this cannot be transmitted directly, but must be transformed first, a modulator changes the waveform to an analog signal using a modulation technique, such as amplitude shift keying (ASK), frequency shift keying (FSK), etc. This makes the signal transmittable and can, depending on the modulation type, increase energy-efficiency, robustness to inference, transmission range, etc [72]. When the signal is received by a receiving node, it has been distorted by numerous physical effects and, therefore, has to be filtered and regenerated first. Finally, it is transformed back into a binary stream using a demodulator and forwarded to the next higher layer, i.e., the MAC layer. In case there are any errors in the signal, these can either be corrected in the PHY during filtering, or later in higher layers.

2.1.2. Medium access control layer

The data link or medium access control (MAC) layer handles the data exchange between two individual nodes in a network. More specifically, it defines a policy of when and how long nodes transmit/receive, what data they send and how they behave when a failure occurs, e.g., retransmit, wait, etc. To this end, it abstracts the raw data stream of the PHY and converts it into data units called frames or packets.

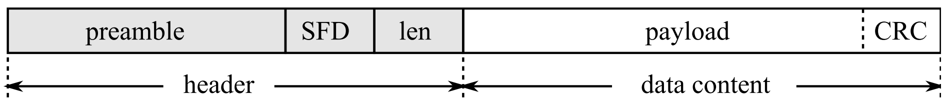


Figure 2.3.: Structure of a packet transmission [4].

Fig. 2.3 depicts the structure of a packet, which consists of 2 parts: a header containing a preamble, a start frame delimiter (SFD) and a length code (len) field, and a data part which holds the payload, i.e., the information (from the upper layers) that needs to be conveyed.

The preamble is required to detect and decode a packet transmission. More specifically, when data is received, the PHY creates a continuous stream of bits, which must be separated into pieces (i.e., packets) and filtered from noise and bit errors. To this end, the receiver continuously searches for the preamble, a special sequence of bits which can be easily differentiated from noise and other transmissions. It indicates the start of a packet and contains information about phase/frequency offsets needed by the demodulator for decoding. After the preamble comes the SFD field, another pre-defined sequence of bits that marks the end of the preamble and the start of the length code, a field storing the remaining data bytes of the packet. Note that preamble, SFD and length code are the minimum required overhead contained in *every* wireless data transmission, independent of its type. This means that they are equally included in data packets, acknowledgments, beacons, and all other transmissions. However, the size and content of these fields depend on the modulation type and other parameters, such as environmental conditions, etc. — more information in [84].

The second part of a packet is the payload, a data field containing the actual information that needs to be conveyed. It is generated by upper network layers and typically contains further structural fields, such as network status, packet priority, etc. However, at the MAC layer, the content of this data is not known, but only the size of it. Lastly, to improve robustness and to detect and discard faulty receptions, a field for error detection is added, for example, a cyclic redundancy check (CRC).

Beside abstracting data streams, the MAC layer also has the task of controlling medium access in order to guarantee a certain quality of service for data exchange such as reliability, energy efficiency, timeliness, etc. This can be achieved by various methods, for example, back-off mechanisms, acknowledgments, carrier sensing and many more. Which strategies are used strongly depends on the MAC protocol itself, the environment, the application used, etc. For example, a video transmission system will use different methods than an ultra-low-power heart-rate sensor that is implanted in a body. A more detailed description of the possibilities and limitations of the MAC layer can be found in Section 2.3 and an overview of state-of-the-art MAC protocols in Chapter 3.

2.1.3. Network and host layers

The network layer abstracts the point-to-point (i.e., node-to-node) connection from the MAC layer into a multi-node network by adding functions such as routing, addressing and traffic shaping. Routing refers to the transmission of data over several intermediate nodes, i.e., these receive packets and retransmit them to the next node. Each retransmission, also referred to as hop, extends the range of the network so that nodes that are not in range of each other can now communicate. Addressing, on the other hand, describes the mapping (translation) of physical addresses in logical ones or names. For example, in computer networks, this describes the step to replace a PC's hexadecimal MAC address by an IP address. Lastly, the networking layer also supports traffic shaping mechanisms, i.e., it can define and manipulate the payload of packets. For example, if several nodes only have to send very small amounts of information, e.g., just a few bytes, these can be combined into one larger packet during routing. This reduces transmission numbers and, therefore, decreases the overhead caused by packet headers, etc.

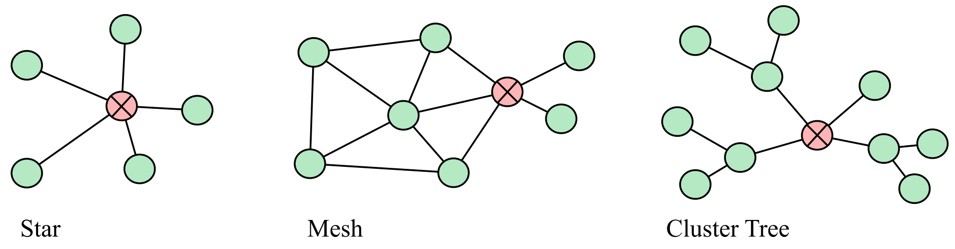


Figure 2.4.: Three common WSN topologies: Star, mesh and cluster tree. Sensor nodes are depicted as green circles and sinks as crossed red circles.

Fig. 2.4 shows three common WSN topologies, i.e., how the nodes are connected within the network. In a star network, sensor nodes transmit their data to the sink in a single-hop fashion, whereas in a mesh network, data can be forwarded over multiple hops to the sink. Similarly, the cluster tree topology is a mixture of mesh and star, where simple sensor nodes transmit their data to more elaborate nodes with forwarding capability. These so called cluster heads then aggregate the sensors' data and transmit it to the sink. The task of the network layer hereby is to coordinate data aggregation, addressing and forwarding in order to allow for efficient communication within the network.

The remaining upper levels, i.e., the host layers (see again Fig. 2.1), stepwise transform the packet-based communication into a high-level flow of information using sessions, function calls and other abstracting methods. This allows applications to exchange information without requiring knowledge about the underlying infrastructure, such as network topology, node density/location, etc. The topmost layer, i.e., the application layer, directly connects to the application running on the node, for which it provides an application programming interface (API) to enable data exchanges.

So far, the previous sections discussed the different steps of communication using the OSI layer model. The next section now further examine the challenges of wireless communication.

2.2. Challenges in wireless communication

In contrast to wired system, in which devices are connected via cables, wireless nodes communicate by emitting electromagnetic waves into the air. Since this is a totally different and typically more intricate transmission medium, new challenges and problems arise [84]. To better understand these, the following section provides an overview of the most prominent challenges that have a major impact on MAC protocol design.

2.2.1. Data corruption and packet loss

A major problem in wireless networks is data corruption, i.e., when signals have been altered such that the receiving node cannot decode them anymore. This is mainly caused by the open transmission medium (air): Whenever a node has data to transmit, it radiates an electromagnetic signal that can be received by all other devices within a certain range. Conversely, this also means that a node can be easily interrupted by other nodes, for example, if these transmit at the same time. The sink node then receives a superposition of multiple signals, which typically cannot be decoded anymore, hence, data is lost [97]. Unfortunately, these simultaneous transmissions are common, since nodes are generally not aware of each other. In addition, nodes cannot detect if they disrupt other nodes, because they cannot receive and transmit at the same time — this is due to very large differences between transmitted and received power, generally in the order of several magnitudes [83].

There are further reasons for data corruption and packet loss. For example, packets can be corrupted by external interference, i.e., by noise from outside the network generated by neighboring wireless systems. In addition, physical effects such as fading, shadowing, etc., can alternate data transmissions such that the cannot be decoded anymore. This occurs, in particular, if the communicating nodes are located far away from each other, i.e., when the received signal is relatively weak [84]. While some of these negative effects can be suppressed easily, e.g., data loss due to fading can be avoided by locating nodes closer to each other or by increasing the transmission power, others are more difficult to address. As result, to enable reliable communication, special mechanisms must be implemented by a MAC protocol to mitigate these negative effects. More on this later.

Lastly, note that packet collisions do not necessarily lead to data loss, but there is a certain chance that the strongest packet can be recovered. This physical phenomenon is called capture effect and typically occurs in networks using frequency-based modulation types such as frequency shift keying (FSK), etc. [27] [31]. While this effect increases the reliability, it is also highly stochastic and, therefore, typically does not lead to reproducible results [31]. Note that the proposed protocols in this thesis neglect capture effect for the reasons mentioned in Section 1.3, however, there are other approaches that do use it. For more information about these protocols, see Section 3.2.1.

2.2.2. External interference and clock drift

External interference describes noise on the communication channel that negatively influences a network by corrupting packet transmissions, etc. It typically originates from neighboring networks, for example, devices using WiFi, Bluetooth, etc., or other sources generating electromagnetic emissions, e.g., sparks in a rotating electric motor. Since external interference is often highly stochastic, it leads to non-reproducible errors that are generally hard to avoid. However, there are some possibilities to mitigate negative effects, for example, change the PHY to use a more robust modulation scheme [51] or change to a less-noisy frequency channel [84]. Further, the MAC layer can implement carrier sensing to detect and avoid interference pulses or the protocol overhead can be lowered, e.g., by using unidirectional communication, to reduce the active time of a node and, therefore, the chance of being disrupted. However, more on this later in Section 2.3.

Another common phenomenon that can negatively influence a network's performance is clock drift. This effect describes the normally occurring variance in a node's crystal oscillator caused, for example, by fabrication-induced variability or temperature changes. Many MAC protocols, in particular, synchronous approaches require nodes to have a common time base and rely on their ability to accurately wait certain time intervals. However, clock drift causes nodes to count time at different rates resulting in longer or shorter waiting times. This can lead to larger collision numbers due to violated schedules, or increased energy consumption, since synchronization must be repeated more frequently.

2.2.3. Mobility, hidden terminals and other problems

In contrast to wired systems, where devices are connected via inflexible cables, wireless nodes are highly mobile and can move freely within a certain from each other. This flexibility enables many interesting applications, however, it also poses a number of challenges and problems.

In particular, environmental conditions can change during movements, for example, nodes might lose connection if they move behind objects or out of range. In addition, in many mobile networks, nodes can enter and leave the network at arbitrary points in time. This makes it difficult to coordinate the network and requires additional effort for registering and de-registering nodes. For example, in intelligent crossroads [55], new cars periodically arrive at the intersection whereas others leave.

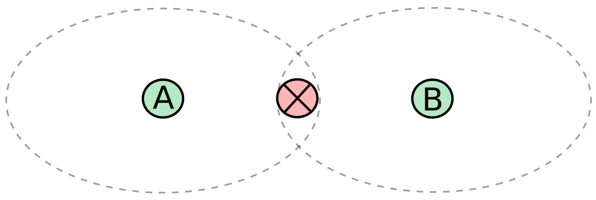


Figure 2.5.: The hidden terminal problem: Node A and B are not within range of each other (dashed line) and, therefore, cannot detect each other's transmissions. As a result, data can be lost at the sink (red circle with cross).

Another common problem is wireless networks are hidden terminals, i.e., when nodes cannot detect each other's transmission although they implement carrier sensing — see Fig. 2.5. This happens if two (or more) nodes A and B that are not in each other's range (dashed lines) try to communicate with a sink that is located between them. Since neither A nor B can detect the other one's transmission, these overlap at the sink and are corrupted. However, hidden terminals can be avoided by locating nodes correctly — this is often not feasible — or by sending a request message before trying to transmit the packet [4], which, however, increases the protocol's overhead. Note that unidirectional networks do not experience this problem, since nodes do not perform carrier sensing.

In summary, there are many more challenges and problems with wireless communication. While some of them can be solved by careful network design, e.g., interference can be mitigated by selecting a less-noisy frequency channel, other problems are more difficult to solve, e.g., internal network collisions. To this end, special mechanisms need to be implemented by a MAC protocol to ensure correct network operation. More on these in the next section.

2.3. Possibilities and limitations of MAC protocols

So far, the previous sections have discussed the challenges and problems of wireless communication. This section now examines the possibilities (and limitations) of a MAC protocol to address these.

2.3.1. Collision avoidance

The strongest negative impact on performance is typically caused by packet loss, i.e., when data is corrupted such that the sink cannot decode it anymore. While this is a result of various effects such as external interference, etc., it is mostly caused by simultaneous transmissions within the network [97]. That is, nodes typically transmit on the same frequency with a relatively high power compared external noise, hence, creating a strong source of interference. Now, in order to suppress this interference, an intuitive approach is to control network access, such that only one node can transmit at a time. For example, with TDMA, each node transmits in a dedicated time slot to avoid collisions. This enables very efficient and highly reliable communication, however, nodes must periodically receive beacon messages for synchronization, which requires additional energy and worsens complexity — more on this in Section 2.5.1.

Another widely-used approach is carrier sensing, where nodes listen for signals on the channel before trying to transmit. If the channel is free, they transmit their data, whereas if the channel is blocked, i.e., another transmission or an external interference pulse was detected, they cancel their packet. This avoids interrupting ongoing transmissions and prevents losing the node's own packet, which, due to the blocked channel, would have been lost anyways. In general, carrier sensing is a very effective and popular method to reduce packet loss that is implemented by many existing MAC protocols, such as CSMA — this protocol is further discussed in Section 2.5.2.

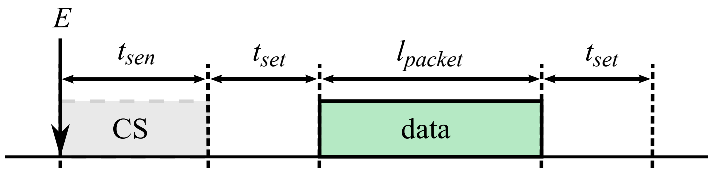


Figure 2.6.: Illustration of a successful packet transmission with carrier sensing

Fig. 2.6 shows the working principle of carrier sensing in more detail. Whenever a node is triggered by an event E and wants to transmit data, it first senses the channel for t_{sen} time. Note that the length of t_{sen} is determined by the protocol or the node's transceiver IC [83]. In case there was no channel activity, the node switches to transmit mode to send its data and then switches back to receive mode to be ready for the next event, e.g., for receiving an acknowledgment. In case the channel is blocked, the node directly goes to sleep after t_{sen} and no data is transmitted. Lastly, note that mode switches are required, since nodes cannot send and receive at the same time. Instead, they must activate their receiver or transmitter module within the transceiver IC, which requires t_{set} time [83].

Although carrier sensing greatly reduces the chance of collisions, it cannot fully prevent them. For example, if it happens that two nodes start carrier sensing at the same time, they cannot see each other, but will assume that the channel is free. Their packets consequently collide — see also Appendix B. Similarly, there may also be hidden terminals that cause collisions. Lastly, note that carrier sensing can only detect that there is activity on the channel, but it can neither determine what is causing it, e.g., another node or external interference, nor how many nodes, etc. are involved. As a result, further mechanisms are required to improve the robustness of a MAC protocol, for example, a retransmission scheme, as discussed next.

2.3.2. Backoff and retransmission schemes

While the previous chapter examined mechanisms to avoid collisions, this section discusses how to react when these have happened. To this end, the concepts of backoff, retransmission and acknowledgment (ACK) schemes are explained and evaluated.

After transmitting a packet, a node does not know whether its data was received successfully or not. For this, a separate notification message is required, called an acknowledgment (ACK). This is realized by sending a small packet containing the address of the node after receiving its data. As a result, if a node receives an ACK, it know that its data has been transferred successfully and can then go to sleep mode. However, if no ACK is received after some time, the node realizes that there was an error and can react accordingly, e.g., by retransmitting the data.

Fig. 2.7 depicts a successful packet transmission with carrier sensing and ACK — note that carrier sensing and ACK are generally used together [4] [6] [21]. Whenever a node is triggered an event E, it first senses the channel and, if it is free, it switches to transmit mode to send its data packet. After transmission, the node switches to receive mode to listen for the ACK and goes to sleep mode after receiving it. If no ACK is received, the node can, for

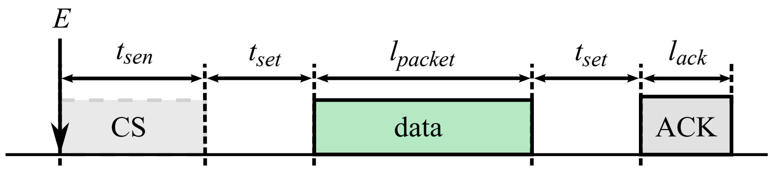


Figure 2.7.: Illustration of a successful packet transmission with carrier sensing and ACK

example, perform a backoff as discussed next. Otherwise, after receiving the data packet, the sink switches to transmit mode to send the ACK and then switches back to receive mode to be able to receive further packets. Note that this last mode switch of the sink is not included in Fig. 2.7, since, due to the carrier sensing mechanism, there cannot be any transmission during that time — more in Appendix B.1.

In summary, acknowledgments are an effective method to detect transmission errors. On the other hand, they also increase complexity and require additional energy and time, therefore, increasing the chance of being interrupted, for example, by external interference — more in the next section. However, ACKs play an important role for many other mechanisms, in particular, for backoff and retransmission schemes. More on these next.

Now, in case there was a collision and a node did not receive an ACK, it has two possibilities: (i) it stops transmitting and goes to sleep mode or (ii) it tries to send its data again. While method (i) can be useful in very energy-restricted networks, most MAC protocols choose option (ii) and try to retransmit the lost packet to increase communication reliability. However, retransmitting a packet directly after the collision will most-likely result in another collision, since other nodes will do the same. To avoid this, each node has to wait for a specific time, called backoff time, before trying to retransmit. Now, there are several strategies to determine backoff times, for example, with CSMA (see Section 2.5.2), these are random, whereas with DEEP (see Section 4), these have a constant, predefined value for each node. In addition, each MAC protocol also specifies a maximum number of retransmission, i.e., how often this process is repeated before a node finally aborts. In general, the larger the retransmission numbers and backoff times, the smaller the collision probability, however, at the costs of increased energy consumption and delays [34].

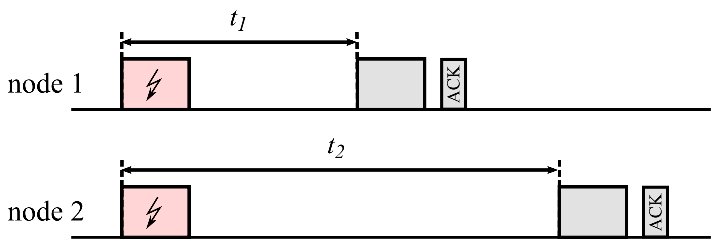


Figure 2.8.: Illustration of a backoff and retransmission scheme

Fig. 2.8 illustrates the working principle of a retransmission scheme for the case of two nodes. Here, node 1 and 2 start transmitting at the same time leading to a collision of their packets. As a result, they do not receive an ACK and try to retransmit their data after waiting for t_1 and t_2 time. Since these times are different for both nodes, their packets do not collide again and they can successfully deliver their data.

In summary, most bidirectional protocols deploy carrier sensing and retransmission schemes to increase the reliability of the network [97]. However, this also adds complexity and overhead to the protocol, and might not be necessary for every application. In particular, some applications require to have a low complexity to reduce costs, e.g., by employing cheap hardware [14]. For this, unidirectional communication can be used, which is described next in more detail.

2.4. Uni- vs. bidirectional communication

Unidirectional communication describes a very basic communication form, in which data can only be transmitted in one way from source to sink. That is, nodes can either send *or* receive and, hence, allow for simple data transfers without feedback only. More elaborate mechanisms, such as retransmission, routing, etc. are not possible, which is both beneficial and disadvantageous as discussed in the following.

Using unidirectional communication can either be a result of physical effects, such as non-symmetric links, i.e., when data can only be conveyed in one way due to obstacles or special environmental conditions [88], or because of hardware limitations, e.g., when nodes are not equipped with a transceiver, but only with a simple transmitter or receiver to reduce costs [14]. However, it can also be a design choice to switch standard bidirectional nodes to receive or transmit-only mode [10]. Although costs will not be as low as with purely unidirectional devices, it adds some flexibility, since nodes can switch back to bidirectional operation, for example, to exchange configuration data.

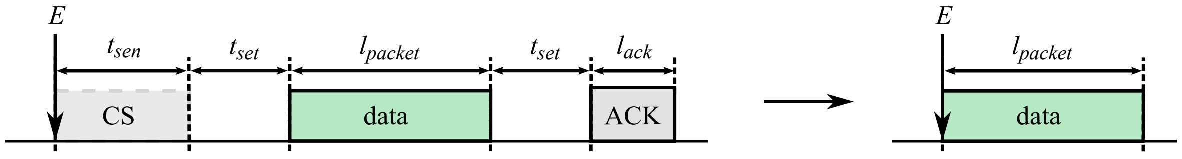


Figure 2.9.: Illustration of a bidirectional packet transmission with carrier sensing and ACK on the left and a unidirectional transmission on the right.

Let us now discuss the advantages of unidirectional communication in more detail. To this end, Fig. 2.9 depicts the different steps required to transmit a packet in a bidirectional network with ACK and carrier sensing (on the left) and in a unidirectional network (on the right). In the case of the bidirectional system, a node first senses the channel for t_{sen} time to check whether it is free to transmit. In case it is free, it switches its transceiver from receive to transmit mode to be able to send the packet, which requires t_{set} time including processing delays, etc. After the packet is sent, the node then switches back to receive mode and listens for the ACK. In summary, a packet transmission requires $t_{sen}+l_{packet}+l_{ack}+2t_{set}$ time, whereas the unidirectional node, which cannot switch to receive mode, just needs l_{packet} time. To emphasize how much larger the overhead for a bidirectional system typically is, let us now consider an example, where nodes are equipped with the commonly-used transceiver CC2420 [82]. This chip offers a transmission speed of 250 kbit/s, requires 192 μs for each mode switch and 128 μs for carrier sensing. The total overhead generated equals a packet transmission of 12 bytes for mode switching and 4 bytes for carrier sensing. Now, if we assume a packet length of 14 bytes and an ACK size of 8 bytes, i.e., both small values that are common in many sensor networks such as home automation, the bidirectional packet transmission requires 1.2 ms in total compared to 0.4 ms for the unidirectional one — roughly three times as long. Clearly, these values strongly depend on the transceiver speed, transmission rate, packet sizes and many other factors and, therefore, vary with different settings. However, even for fast transceivers, i.e., those with short switching times, ACKs, carrier sensing, etc. still add considerable overhead [94].

The larger overhead of bidirectional communication results in several disadvantages: First, it increases the energy consumption of a node, as the transceiver must be active for a longer time. Second, this creates a larger amount of traffic, not only because an ACK must be transmitted, but also because the sink node — which must also switch modes during a packet reception — cannot listen for other packets while switching modes; these are then lost or corrupted. In particular, in networks with high data traffic and/or external interference, the additional overhead leads to increased collision numbers and reduced performance, as analyzed later in simulation in Section 6.4.

Lastly, bidirectional networks typically have a much higher complexity compared to unidirectional systems, which cannot implement elaborate feedback and control mechanisms. As discussed before, using unidirectional nodes consequently allows using simpler hardware such as less-powerful processors, smaller batteries, etc. and can, therefore, strongly decrease costs [14]. For example, in home automation, a simple unidirectional light switch from Intertechno[1] costs around 15 euros, whereas the price of its bidirectional equivalent from Z-Wave[2] is around 50 euros. Considering that many networks have high numbers of nodes, e.g., typically around 30 to 50 devices for home automation, using unidirectional nodes can result in considerable cost savings.

However, there are also several disadvantages when using unidirectional communication, in particular, reliability is decreased; no carrier sensing or synchronization is possible, hence, it is difficult to coordinate transmissions. As a result, most existing protocols cannot be used, as these rely on such feedback mechanisms. Instead, special MAC protocols need to be implemented, such as DEEP and RARE. However, before explaining these, the following sections first cover the current state-of -the-art of MAC protocols, starting with the widely-used (bidirectional) protocols TDMA and CSMA.

2.5. Major wireless MAC protocols

This section describes the widely-used protocols TDMA and CSMA, which are de-facto standards for WSNs and used in a large number of applications [28] [33] [52]. Note that these will later be simulated and compared to DEEP and RARE in Section 6.4.

2.5.1. TDMA

Time Division Multiple Access (TDMA) is a MAC protocol designed for high data rates and low delays. It is mainly used for mobile telephony and internet (e.g., GSM) for which it is defined by multiple standards such as 802.16 (WiMax) [1] and DECT [2]. However, TDMA is also often used for WSNs, especially for those requiring high data rates and low delays [99].

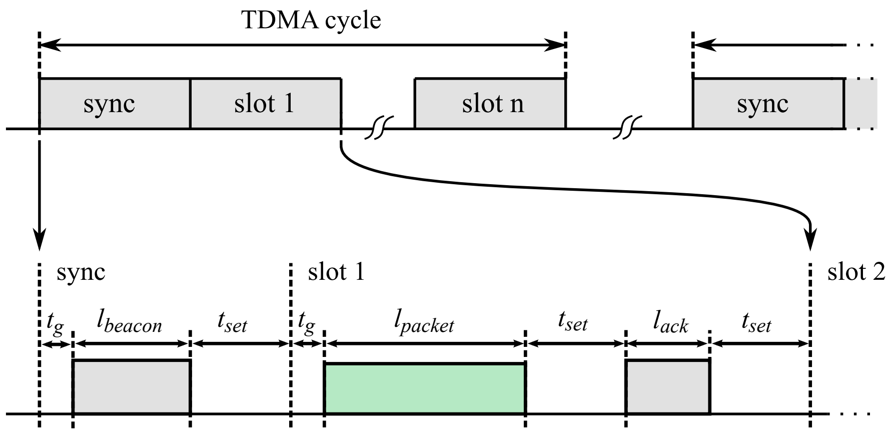


Figure 2.10.: Illustration of TDMA's slotted communication structure.

In TDMA, communication is organized in cycles as shown in Fig. 2.10. Each cycle starts with a beacon message for synchronization, followed by number of dedicated time slots in which nodes can transmit. In classic TDMA, each node has its own slot allowing for a

[1]www.intertechno.at
[2]www.z-wave.com

collision-free transmission. However, there are other approaches in which slots are either shared [65] or assigned dynamically [18] [96] — see also Chapter 3. Once a cycle has finished, i.e., all slots have been processed, another identical cycle starts directly or after a short waiting time.

Whenever a node wants to transmit data, it waits for its slot to begin and then directly transmits its packet. To this end, typically no carrier sensing is used, since slots are generally collision-free. However, many applications implement an ACK scheme to be able to retransmit data and, therefore, increase robustness against external interference [65] [96].

Note that each slot starts with a guard time interval t_g to allow for slight clock inaccuracies. That is, nodes might drift apart by some time before resynchronizing and, if that drift becomes too large, they might violate their slot boundaries and collide with nodes from neighboring slots. To prevent this, both the beacon rate and t_g must be adjusted correctly. In general, the higher the beacon rate, the shorter the time that nodes can drift apart before resynchronizing and, therefore, t_g can have a smaller values. This lowers delays, however, it also increases energy consumption, since nodes must resynchronize more frequently. Conversely, larger t_g allow lowering the beacon rate, but at the costs of higher delays. How to configure this correctly depends on the application, i.e., how much energy is available, what oscillators are used, etc. — an example can be found in Section 6.2. Note that the sink sends a beacon in every cycle, but sensor nodes only listen for them if they need to synchronize, i.e., at the configured beacon rate.

In summary, TDMA enables very high data rates and very low delays due to its slotted, collision-free structure. However, it also suffers from high complexity and communication overhead, resulting in typically higher costs. Moreover, efficiency is only good at high traffic loads, since during lower traffic, many slots are empty, while the overhead remains the same. And lastly, TDMA is rather inflexible, since assigning nodes to slots requires several control packets to be exchanged. To mitigate these problems, many alternative and/or modified versions of TDMA have been presented in the literature, as discussed later in Chapter 3.

2.5.2. CSMA

Carrier Sense Multiple Access (CSMA) is a MAC protocol designed for low complexity, costs and high flexibility. It is used in many prominent applications such as WiFi and Zigbee and defined by multiple standards such as 802.11 [3] and 802.15.4 [4]. For WSNs, in particular, the 802.15.4 standard is used [16] [59] [94], which defines two CSMA-based protocols: a hybrid superframe MAC (in slotted, beacon-enabled mode), and classic CSMA (in unslotted mode). While the first one is designed for applications that require low latencies and a guaranteed bandwidth, the second one offers low complexity and overhead, similar to DEEP and RARE. For this reason, only unslotted CSMA is considered for the rest of this work.

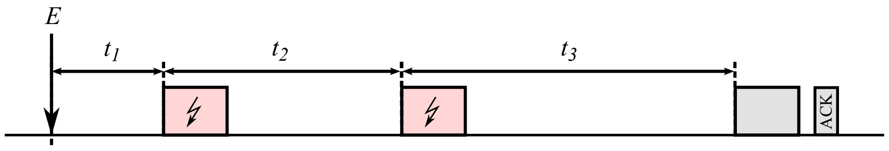


Figure 2.11.: Working principle of CSMA: A node performs a random backoff before every transmission including the first one. If the transmission fails, the node performs another random backoff and tries to retransmit, whereby the maximum possible backoff time becomes larger every time.

Fig. 2.11 depicts the working principle of CSMA as per 802.15.4. Whenever a node is triggered by an event E, it first performs a backoff and then tries to transmit. Before transmission, carrier sensing is used to assess the channel state and the node only sends its packet if the channel is free. In case the channel is blocked or whenever a transmission failed, i.e., no ACK was received, the node performs a random backoff before trying to retransmit. The backoff time is calculated using the binary exponential backoff formula $\text{rand}(1, 2^{BE} - 1) \cdot t_{base}$ with BE being the backoff exponent. The value of BE is increased by 1 every time a transmission fails, however, it is limited by the contention window defining its minimum and maximum value — by default this is $minBE = 3$, $maxBE = 5$ [4]. As a result, the backoff time becomes larger every time a transmission fails, therefore, reducing the chance of a subsequent collision.

In summary, CSMA is a very flexible protocol with low complexity and overhead. However, it suffers from reduced performance under high traffic loads, as collision numbers and delays increase considerably. In addition, CSMA needs to be configured correctly, since performance is typically low with default parameters [9], which, however, is very challenging due its complex model [34]. More on CSMA can also be found in the next chapter, where further state-of-the-art protocols are reviewed.

Chapter 3.

Related work

The wide variety of WSN applications with their individual requirements and characteristics has led to a plethora of different MAC protocols. This chapter provides an overview of the most common types and strategies and further summarizes those protocols that are comparable to the ones presented in this work.

3.1. Classification of MAC protocols

In order to categorize the vast amount of different MAC protocols, several classification methods have been presented in the past. A widely-used approach is to categorize protocols according to their working principle in contention-based and contention-free protocols [28] [80]. Contention-based protocols react to contention (e.g., collisions, blocked channels, etc.) and try to resolve it. For example, CSMA is a contention-based protocol that reacts to collisions or blocked channels by performing backoffs and retransmissions. Contention-free protocols, on the other hand, try to avoid collisions, etc. in the first place by implementing mechanisms such as synchronization. For example, with TDMA, nodes transmit in dedicated time slots to avoid interference with other nodes.

Another classification described in [48] divides MAC protocols in synchronous, asynchronous and hybrid techniques, i.e., according to their timing behavior. In synchronous networks, e.g., TDMA, nodes share a common time base by periodically adjusting their clocks to each other. This makes it possible to avoid collisions by, for example, creating schedules in which nodes transmit at dedicated time instants. Asynchronous protocols such as CSMA, on the other hand, do not synchronize nodes, but transmissions are uncoordinated and may occur randomly. Lastly, hybrid approaches combine methods both from synchronous and asynchronous communication, for example, switch between CSMA and TDMA depending on the network traffic. This allows the system to adapt to changing conditions and use the optimal strategy for improved performance. Note that although this classification method is very similar to the previous, contention-based/free one, there are differences. That is, synchronous protocols must not necessarily be contention-free, but can also be contention-based, e.g., a collision resolution scheme for TDMA [65]. Similarly, asynchronous protocols can also be contention-free, e.g., with persistent CSMA as per 802.11 [3], nodes do not back off if the channel is blocked, but wait until it is free again and then transmit.

Clearly, there are further classification strategies, for example, it is possible to categorize protocols according to their properties, e.g., delay-tolerant and delay-sensitive, loss-tolerant and loss-sensitive etc. [80] In this work, however, only a brief overview of the most common protocols and strategies is provided, hence, the synchronous, asynchronous and hybrid classification is used for the rest of this work. That is, it is universal, i.e., it can accurately classify most protocol types, and is widely used in the literature [28] [47] [48]. In the following sections, the individual categories are explained in more detail and commonly-used MAC protocols are summarized for each category.

3.1.1. Synchronous

In synchronous networks, nodes typically share a common clock by periodically sending beacon messages or by synchronizing to external events. This allows them to efficiently schedule transmissions and sleeping times [35] [52], resulting in high maximum throughput and bounded delays. On the other hand, synchrony comes at the cost of additional energy consumption and complexity, making it less suitable for networks with long idle times or for environments with high levels of external interference, as control messages can be lost as well [99].

The most well-known synchronous protocol is TDMA, which is defined by multiple standards [1] [2] and used by many prominent applications such as mobile telephony and internet (e.g., GSM and LTE) [1]. It consists in nodes sending their data in dedicated time slots to avoid collisions, whereby synchronization is maintained by periodically receiving beacon messages from a central node. This enables a very high channel efficiency of up to 100 % and short delays, however, there are also some drawbacks. In particular, TDMA suffers from decreased efficiency during low traffic load, i.e., when most nodes are idle. In this case, many slots will be empty, which decreases the overall efficiency. In addition, TDMA is rather inflexibility, since assigning nodes to slots requires several control messages to be exchanged. Especially in mobile networks, where nodes can enter or leave the network at arbitrary point in time, this generates considerable overhead.

To solve these problems and improve the performance of classic TDMA, several extension and modifications have been presented in the past. For example, in [65] slots are assigned to multiple nodes to shorten the overall cycle length and reduce the number of unused (wasted) slots. Since this results in possible collisions, e.g., when multiple nodes transmit within the same slot, a collision resolution scheme is presented, which generates a number of additional slots upon collisions to enable retransmissions of lost data. This considerably decreases delay and energy consumption compared to classic TDMA, however, benefits diminish for higher traffic, since many additional slots must be generated to resolve conflicts. Another approach in [96] dynamically adapts its schedule, i.e., slot numbers and times, to changing traffic and network conditions. This allows minimizing idle slots and cycle times, however, it adds additional complexity to the network and is only suited for slowly changing, non-random traffic.

There are further synchronous MAC protocols, for example, in [41] [86], nodes adjust their transmission intervals using prediction algorithms to maximize sleep times and reduce energy consumption. Or in [30], nodes are synchronized so that concurrent transmissions of the same packet do not lead to packet loss, which is particularly interesting for network flooding, i.e., when certain data has to be distributed to all nodes in the network. In summary, these protocols allow for good energy efficiency and/or high throughputs, however, they still have the disadvantages of synchronous communication, i.e., a high complexity and vulnerability to external interference.

3.1.2. Asynchronous

Asynchronous MAC protocols forgo any synchronization, but instead implement mechanisms such as random back-off or retransmission schemes to increase reliability. They offer good energy efficiency and high flexibility, making them ideal for dynamic, low-cost networks. On the other hand, since transmissions are uncoordinated, they incur in higher packet loss rates and a typically unbounded delay, especially for higher network traffic.

A very well-known asynchronous protocol is CSMA, which is used by many prominent applications such as WiFi and Zigbee and defined by several standards such as 802.11 [3] and 802.15.4 [4]. In contrast to TDMA, transmissions are not coordinated, but nodes can send their packets at arbitrary points in time. Since this leads to conflicts with other nodes, backoff, carrier sensing and retransmission schemes are implemented to reduce contention and improve the chance of a successful transmission — see also Section 2.5.2. In summary, this enables low-complex and highly-flexible communication, however, during high traffic, the performance quickly degrades due to high collision numbers — this is further evaluated in Section 6.4.

Another problem of CSMA is that it is general purpose, i.e., it is designed to be suitable for a wide range of applications. While this allows CSMA to be used nearly everywhere, it also means that MAC parameters such as backoff times, packet numbers, etc., are typically not configured correctly for a specific application. Although these can still be adjusted, finding the optimal setting is difficult due to high complexity [34], while without adaption, i.e., with default parameters, performance is often low [9]. To solve this problem, numerous modifications have been proposed, for example, dynamic, learning-based adaption of parameters [16] or a modified backoff scheme [59]. These typically improve the average performance, however, also lead to increased complexity and a lower worst-case performance.

There are many other protocols based on asynchronous communication. For example, in [6], nodes change their data transmission interval depending on how many packets were lost in the past or in [25] packet lengths are adjusted depending on the current network load. In summary, these allow for highly flexible communication at (typically) low complexity, however, the disadvantages of asynchronous communication remain. That is, these typically cannot provide guarantees on performance in the worst case and do not perform well under very high traffic loads.

3.1.3. Hybrid

Hybrid MAC protocols try to combine the advantages of synchronous and asynchronous approaches. This can, for example, be done by switching from CSMA to TDMA under high contention [99] or by creating a super frame, which divides transmission time in synchronous and asynchronous slots [4] [78]. Although these approaches increase the average reliability, they incur additional complexity and are usually limited to the performance of either TDMA or CSMA in the worst case.

A commonly-used hybrid protocol is slotted (beacon-enabled) CSMA as defined in the IEEE standard 802.15.4 [4]. It is based on superframes which divide transmission time into an asynchronous contention-access period (CAP) and a synchronous contention-free period (CFP). In CFP, a small number of registered nodes can reliably transmit data in individual time slots similar to TDMA. In CAP, the more flexible but less reliable classic (unslotted) CSMA is used for transferring data, e.g., for low priority nodes that tolerate slight data loss or new nodes that want to register for the CFP. In summary, this hybrid superframe allows for both highly flexible and reliable data transfers. Nevertheless, the overall complexity is higher compared to standard CSMA and TDMA and, in the worst case, performance is limited to either CSMA (in CAP) or TDMA (in CFP).

Other hybrid protocols implement collision resolution schemes, i.e., mechanisms that separate colliding packets to allow for a collision-free retransmission [43] [61]. For example, with Strawman [61], nodes send a contention packet of random length whenever their data collided and they did not receive an ACK. The receiver then selects the node with the longest packet and transmits a decision message. This process is then repeated until all collisions are resolved. In summary, Strawman allows for fast collision resolution, however, its random nature makes it not suitable for real-time applications. Another protocol, called Stairs [43], tries to speed up the resolution process by using the received signal strength indicator (RSSI) to determine the number of contending nodes and create a schedule. This greatly reduces the number of required contention packets, however, the use of the RSSI channel is highly error-prone and works only for a low number of contenders, as it quickly starts to saturate.

There are many other hybrid MAC protocols in the literature. In summary, these generally improve the average performance of the network, however, at the costs of increased complexity, slower reaction times to events (i.e., longer delays) and decreased worst-case performance [65]. In the following sections, further strategies are presented to improve the performance of a network such as delay, reliability etc.

3.2. Protocol strategies

This section provides an overview of common mechanisms and strategies used by many MAC protocols to improve the quality of service (QoS) of the network. Note that these are independent of the previously discussed classifications of a MAC. For example, the capture effect can be used in both synchronous, asynchronous as well as hybrid networks.

3.2.1. Capture effect and node placement

The capture effect describes the physical phenomenon that there is a certain chance (i.e., a capture rate) that the strongest packet can be recovered in the event of a collision. This effect occurs in all networks that implement frequency-based modulation types such as frequency shift keying (FSK), direct-sequence spread spectrum (DSSS), etc. [36] and can noticeable reduce collision numbers and, therefore, improve energy consumption, reliability, etc. of the network [27] [31]. To increase the capture rate and, therefore, the positive effects, several approaches have been presented in the past. For example, [93] defines a policy for the optimal node placement and [31] analyses how the number of sinks in a network influences the capture rate. Results show that packet loss can be reduced considerably, e.g., by up to 35 % when using three correctly-placed receivers instead of only one [93].

Despite many advantages, the capture effect also has numerous drawbacks. In particular, its highly stochastic nature makes it impossible to perform any worst-case analysis and to provide any guarantees on performance. That is, the capture rate depends on many, often random factors such as node location, which parts of the packets overlap, modulation type and many more [31]. In addition, at most one packet can be recovered in the case of a collision — all other are lost — which limits the maximum performance gain to 50 %. Lastly, fairness [27] is poor, meaning that nodes located far away from the receiver can potentially *starve*, because closer nodes always have higher signal strengths. In summary, these negative effects are the reason that this thesis and many other works [10] [21] [94] do not consider capture effect, but pessimistically assume that packets cannot be recovered in the case of a collision — see also Section 1.3.

Another commonly-used strategy is to place nodes at selected locations to improve the performance of the network, for example, by reducing interference between neighboring nodes or improving the capture rate as discussed above. In [23], the authors came to the conclusion that positioning the sink nodes correctly can greatly reduce the energy consumption, since sensors can reduce their transmission power to reach the sink; this is also called transmission power control (TPC) [89]. However, such strategies are only suitable for static networks where nodes do not move and may not be feasible for applications where node position are already fixed.

3.2.2. Clustering and data aggregation

Clustering describes the strategy of dividing a network into smaller groups of nodes (clusters) [7]. Each sensor then transmits its data to a local sink, called cluster-head, which combines all packets and forwards them to the main sink, see Fig. 3.1. This allows sensors to reduce their transmission power — cluster-heads are typically located closer — improving energy consumption and reducing interference between neighboring nodes and clusters [89].

Further, clustering offloads complexity from sensor nodes to cluster heads, which allows using simpler and, hence, more cost- and energy-efficient sensor nodes. This is exploited in [14] where the authors replace the large number of sensor nodes with cheaper unidirectional ones to save costs. However, there are also drawbacks. In particular, data must always be conveyed via cluster head adding another hop and, therefore, another error source. If the cluster head loses packets, all the previously collected information is lost.

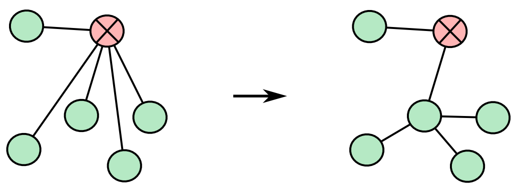


Figure 3.1.: Simplified illustration of clustering: In a regular network (left), nodes transmit data directly to the sink, whereas in a clustered network (right), data is transmitted to a local sink (cluster head) which then forwards it to the main sink.

Another common strategy is data aggregation. It describes the process of collecting and processing data before forwarding it to the sink in order to reduce packet sizes and/or transmission numbers [92]. This method is mainly used in multi-hop networks, such as the previously mentioned clustered systems, however, it can also be used in single-hop networks. That is, sensor nodes can collect multiple samples before transmitting them in a single packet [90]. In general, data aggregation greatly reduces overhead and, therefore prolongs the network (battery) lifetime. For example, in [53], battery life could be extended by up to two times. However, data aggregation typically causes long delays, since data gathering and processing requires time. As a consequence, it is typically only suitable for slower applications such as environmental monitoring [92], but generally not for real-time applications.

3.2.3. Duty cycling

Nodes are typically battery powered and, hence, must economize on energy. A very straightforward approach to reduce energy consumption is duty cycling, i.e., to put nodes into sleep mode and only wake them up for very short moments. The smaller the duty cycle, i.e., the ratio of active time compared to total time, the more energy efficient a network is. On the other hand, a smaller duty cycle also leads to longer inactive (sleep) times in which nodes cannot communicate. For example, a sink cannot receive packets in sleep mode, but these are lost. To avoid this problem, several approaches have been presented in the past that ensure that nodes wake up at the right time. For example, in [98], nodes periodically wake up and shortly listen for incoming data traffic. If they receive a wake-up message, i.e., an empty packet that signals a transmission request, they wake up to receive the following packet. This is a very simple approach with good performance, however, it requires relatively long wakeup messages to be sent — these must be larger than the wakeup interval to be safely detected — prolonging delays and increasing energy consumption. Another approach in [35] tries to avoid these long wakeup messages by synchronizing nodes so that they sleep and wake up according to a schedule. This greatly improves delay and energy consumption, however, it also adds complexity and might not be suitable for larger networks, as synchronization is difficult to maintain.

In general, duty cycling is an effective strategy to reduce the energy consumption of a network and prolong its lifetime. For example, in [13], the authors achieved a six times longer battery lifetime. However, on the other hand, duty cycling is typically only suitable for networks with low event activity, i.e., where nodes do not have to wake up frequently to not generate too much overhead. In addition, it causes large delays making it less suitable for time-sensitive systems.

3.2.4. Miscellaneous

There are many further strategies that aim to improve the QoS of a network. For example, Pereira et al. [70] propose a dominance protocol similar to CAN, in which nodes bitwise transmit their priority before data packets. A logic *1* is dominant and nodes transmit a short pulse, whereas a logic *0* is recessive and nodes listen and drop out if they sense a pulse. This approach is very robust, however, it also incurs in long delays, since nodes must often switch between receive and transmit mode.

Another common strategy is to implement special modulation schemes, for example, the authors in [45] propose a modulation that increases data rate and energy efficiency. However, such approaches are not very practical, since they require special, non-standard transceivers and, therefore, cannot be used with most existing hardware.

Many protocols also implement channel hopping, i.e., a mechanism where nodes periodically switch radio channels to reduce interference. For example, the approach in [81] can greatly improve the packet reception ratio, however, at the costs of a strongly increased complexity and energy consumption. In summary, there are many more strategies, all aiming to improve certain quality aspects of a network. For more information see also [74].

3.3. Related MAC protocols

While the previous sections discussed the different types and strategies of MAC protocols, this section now provides an overview of those protocols that a similar to the ones proposed in this thesis. The focus hereby is particularly on reliable and unidirectional protocols, however, also hybrid and unacknowledged approaches have been included for the sake of completeness. That is, these follow a similar strategy, i.e., they try to reduce communication overhead, complexity, etc., while maintaining a certain QoS.

The two protocols presented in [14] [71] use a hybrid approach, i.e., they are composed of a high number of transmit-only nodes forming clusters for cost reduction and so-called *cluster heads* with reception capability. Cluster heads collect packets from their corresponding transmit-only clusters and forward them to receivers. Since they can acknowledge packets and perform carrier sensing, more sophisticated communication schemes can be implemented upon them. For example in [71], cluster heads use a configurable receiver that only collects data packets complying with a pre-specified signal strength. By this, signals from neighboring clusters can be filtered out reducing interference and collision rates. However, if many cluster heads are used, costs and energy consumption increase rapidly. Moreover, these methods cannot provide any reliability guarantees and packets may potentially never reach their receivers due to collisions, in particular, within a transmit-only cluster.

Another hybrid approach presented in [95] also consists of two sensor node types: low-priority, transmit-only nodes (LP-nodes) and high-priority, bidirectional nodes (HP-nodes). Both node types are triggered periodically and transmit a number of redundant data packets with constant inter-packet times. These times are known by the sink, which uses them to schedule HP nodes to transmit in vacant time slots. As a result, HP-nodes do not collide with LP-nodes and the overall transmission reliability increases. However, this approach again incurs in increased costs and energy usage, since the resulting improvement in reliability strongly depends on the number of HP-nodes. Further, it assumes that nodes are triggered periodically in a known interval, which is not practicable in event-triggered applications. In contrast, the protocols proposed in this thesis are applicable to both periodic and event-triggered scenarios.

Other approaches [37] [38] use backscatter communication to reduce costs and energy consumption of nodes. For example in RFID systems [37], a sink node (reader) transmits a radio signal, which sensor nodes (tags) can use as a power source and as a carrier for reflecting back their encoded data. Similar to classic transmit-only systems, backscattering does not allow tags to detect transmissions from other tags [17]. However, in contrast, the reader's carrier signal allows synchronization or waking up tags with specific IDs. As a consequence, most existing backscattering systems either use variants of Aloha and TDMA or tree search methods that aim to avoid collisions by identifying only one tag at a time. These benefits, however, also come with some major drawbacks. In particular, backscattering is not well-suited for applications such as home automation, where devices have long idle times, e.g., in the order of hours, but upon activation, data must be transmitted timely. Since backscattering is receiver initiated, the sink either has to pull for data periodically, which adds additional delay, or provide a continuous data signal, being especially problematic in (typical) settings with multiple sink nodes.

An unacknowledged MAC for slotted CSMA (as per 802.15.4) has been presented in [94]. Here, nodes use carrier sensing to locally measure busy channel probabilities and to estimate reliability. These values are then used to tune MAC parameters, in particular, contention window sizes are changed to prolong or shorten backoff times. This improves performance and allows reaching the same reliability as acknowledged CSMA, while in return having a lower energy consumption by missing ACKs. However, this protocol is based on a heuristic algorithm to estimate average reliability and cannot provide any guarantees on its worst-case behavior.

In [21], another unacknowledged CSMA mechanism was presented that uses a modified RSSI (received signal strength indication) to obtain additional information about interfering sources and decide whether to transmit or to back off instead. This increases the average performance compared to classic CSMA [21]; however, no analysis framework is provided for assessing the worst-case performance.

In the context of purely unidirectional networks, Zhang et al. presented a protocol taking advantage of capture effect, i.e., it assumes that there is a certain chance (i.e., a capture rate) that the strongest packet can be recovered in the event of a collision — see also Section 3.2.1. To maximize the capture rate, the authors present policies to determine the optimal number of receivers in a network as well as their (physical) placement. Results show that packet loss can be reduced by 35 % when using three correctly-placed receivers instead of only one. However, this approach has a number of drawbacks, for example, it is not well suited for mobile networks, as receivers would have to be relocated continuously for good performance. In addition, capture effect causes poor fairness [27], meaning that nodes located far away from the receiver can potentially *starve*, because closer nodes always have higher signal strengths. Lastly, in contrast to the proposed protocols DEEP and RARE, this protocol is unable to provide any kind of worst-case guarantees due to the highly stochastic nature of capture effect [31].

Another unidirectional approach called Timing Channel Aloha (TC-Aloha) has been proposed by Galluccio et al. [32]. This scheme encodes parts of the information in the inter-packet separation of the different nodes. As a result, the packet length and transmission time can be reduced, thus, decreasing energy consumption and the probability of packet collision. In order to improve reliability, packets are transmitted multiple times. Clearly, to recover the information embedded in the inter-packet separation, at least two packets must be received. By an analytical framework and experimental results, Galluccio et al. [32] prove that TC-Aloha increases data throughput; however, no guarantees can be given on whether data always reaches the corresponding receivers on time.

In [87], Weißenhorn and Hirt proposed a probabilistic approach similar to RARE that allows quantifying the collision probability of periodic data packets. However, the analysis is limited to the collision probability of individual data packets and does not consider the probability of loosing a given number of consecutive packets, which, however, is more common [32]. As a result, [87] can only be used by a small set of applications.

In [19], Cardell-Oliver et al. evaluated three different error control strategies typically used in unidirectional networks: temporal diversity, spatial diversity and code-based methods. In temporal diversity, data packets are, for example, transmitted repeatedly at different times, while spatial diversity aims to separate devices geographically such that they do not fall within each other's range and, hence, cannot interfere with one another. Code-based methods add redundant data, which can be used by the receiver to correct damaged packets. However, the increased packet length leads to less energy efficiency and higher collision probability due to longer transmission times. Cardell-Oliver et al. further concluded that temporal and spatial diversity generally yield better results than code-based strategies, in particular for indoor scenarios [19]. Note that both DEEP and RARE make use of temporal diversity, i.e., they transmit data at selected point in time. By contrast, spatial diversity was intentionally omitted due to its limited flexibility — see Section 3.2.1.

Andersson et al. [10] presented a unidirectional protocol similar to DEEP that can guarantee that data always reaches its destination in the worst-case. To this end, each node transmits its data as sequences of redundant packets with constant inter-packet times. In contrast to DEEP, inter-packet times are selected via ILP (integer linear programming) minimizing the transmission durations of all sequences of packets. However, due to the high complexity of the problem, patterns for only a small number of nodes can be found in an acceptable time. To address this concern, Andersson et al. presented an alternative algorithm that heuristically searches for transmission patterns [11]. Although this second algorithm is considerably faster than their ILP-based approach, it is still time-consuming as shown later in an experimental evaluation in Section 6.3 and in simulation in Chapter 6.4.

In this thesis, two MAC protocols are presented that enable highly reliable communication in unidirectional WSNs. In contrast to the aforementioned solutions, these do not pursue a best effort approach, but provide frameworks that allow calculating their expected worst-case performance. To this end, node specific parameters and practical factor such as external interference are regarded during calculations, resulting in a typically better performance than fixed-parameter MACs, such as CSMA. The presented approaches are general and can be applied to a wide variety of different applications using commonly available hardware. More on this in the following chapters.

Chapter 4.

The DEEP protocol

The Deterministic Protocol (DEEP) is designed to enable fully reliable communication in low-cost, unidirectional networks, i.e., it can guarantee that data always reaches its destination in the worst case (without interference, more on this later). Originally, it has been designed to be used in home and building automation, an area that particularly strives for lost costs to be attractive for consumers. These networks mainly consist of sensors that solely report data to a sink for which no external control or feedback are needed [95] and, hence, allow unidirectional nodes to be used. The DEEP protocol enables reliable communication, which is required by some safety-critical devices, such as heating controllers or security; these were typically based on bidirectional communication previously to avoid unreliability issues. Note that DEEP can also be used for many other applications, in particular, those that require both high reliability and low complexity. An overview about exemplary application areas can be found in Section 7.1.

Before examining the working principle of the DEEP protocol, let us shortly recall the underlying models and assumptions, as discussed previously in Section 1.3. That is, the network consists of n transmit-only nodes and multiple receive-only sinks that are connected in a single-hop (star-topology) fashion. Data transfers can either be triggered by events or occur periodically, and at least one packet must arrive within a deadline d at the sink to ensure normal (error-free) operation. Lastly, it is assumed that packet loss originates from simultaneous transmissions, whereas external interference is neglected[1] and analyzed separately.

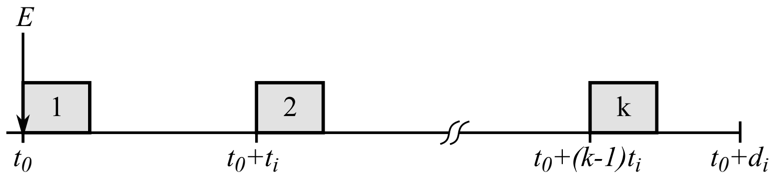


Figure 4.1.: Working principle of DEEP: After being activated by an event E, node i starts transmitting a sequence of k_i redundant data packets with constant inter-packet times t_i within a deadline d_i.

Fig. 4.1 depicts the working principle of DEEP in more details. Whenever a node i is triggered by an event E, it starts transmitting k_i redundant packets with constant inter-packet times t_i with $0 \leq k_i \leq n$, $0 \leq i \leq n$ and $t_i \in \mathbb{R}_{>0}$. Each packet has a length of l_i and must be transmitted before a deadline d_i; both parameters depend on the type of node, e.g., a temperature sensor usually requires more data to be sent than a simple light switch, which, however, must transmit its data faster, i.e., within a shorter deadline. Now, there are several different parameters, which must be considered during protocol design. In the case of the network size n, deadline d_i and packet size l_i, these are determined by the application

[1]Note that applications such as home automation are typically deployed indoors and, therefore, well-shielded from outside by walls. In addition, interference can be reduced considerably by choosing a robust modulation scheme or a less-noisy frequency channel, as mentioned previously in Section 2.2.2.

and are known in advance. The packet number k_i and inter-packet times t_i, on the other hand, must still be selected. To this end, the next section introduces a mathematical model to calculate these values.

Lastly, note that the DEEP protocol has been previously published in [68] and [69]. More specifically, [69] contains the base model from Section 4.1, whereas [67] published the extended model with arbitrary node types from Section 4.2.

4.1. Analytic model

This sections gradually derives a mathematical framework to calculate the missing network parameters k_i and t_i. To this end, a simplified system with only one node type is considered in order to reduce complexity and facilitate understanding. One node type means that all nodes in the network are the same, i.e., they have the same packet length and deadline (denoted by l_{max} and d_{max}). Although this seems to be a very restrictive assumption, such networks are quite common. That is, many WSNs only monitor a few environmental conditions for which one node type is sufficient, e.g., a weather network usually consists of temperature/humidity sensors only. In case this is not possible, but there are different types, these can be converted into one type by pessimistically assuming the shortest deadline and longest packet length for them, i.e., $d_{max} = \min\limits_{i=1}^{n}\{d_i\}$ and $l_{max} = \max\limits_{i=1}^{n}\{l_i\}$. For example, if we have a temperature sensor that needs to transmit 24 bytes data within 5 minutes and a light switch with 8 byte data and 500 ms deadline, these can be *safely* converted to one node type with $d_{max} = 500\,\text{ms}$ and $l_{max} = 24\,\text{bytes}$.

The above assumption has the advantage that complexity is greatly reduced and thus an analytical (mathematical) model can be used. As we will see later, different node types with arbitrary deadlines and packet sizes increase complexity considerably and, hence, require a heuristic search algorithm instead. Although this has several benefits as discussed later, the analytical framework allows calculating performance and parameters via formulas in considerably less time and at much lower computational complexity.

4.1.1. Selecting packet numbers

For a reliable communication, it has to be guaranteed that at least one packet reaches its corresponding receiver within a pre-specified d_{max}. Towards this, we first obtain the worst-case number of packets that need to be transmitted by any node in the system. This is stated in the following lemma.

Lemma 1. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k_i packets within d_{max} where $d_{max} < t_{act}$. If $t_{act} \geq 2 \cdot d_{max}$ holds, it cannot be avoided that at least $n-1$ packets out of k_i be lost independent of the inter-packet time t_i of the different nodes.*

Proof. Nodes are independent of each other and can be triggered at arbitrary points in time. It might be the case that every time a node tries to transmit a packet, this gets interfered by a packet of another node being activated at that time independent of the inter-packet time t_i of the different nodes. As a result, the corresponding packets are lost.

Let us assume that node j starts transmitting at time t_0. By assumption, this then sends k_j packets within $[t_0, d_{max}]$ with a constant inter-packet time t_j. In the worst case, this node sends its last packet at time $t_0 + d_{max} - l_{max}$ such that this packet is fully transmitted by $t_0 + d_{max}$. If node i starts transmitting its k_i packets at time $t_0 + d_{max} - l_{max}$, the last packet of node j and the first packet of node i will be lost. If the remaining $n-2$ nodes in the system get activated at $t_0 + d_{max} - l_{max} + \alpha_i \cdot t_i$ where α_i is an integer number and $1 \leq \alpha_i \leq n-2$, all first $n-1$ packets of node i will be lost independent of any inter-packet time in the system.

Now let us further assume that the n-th packet of node i is sent at $t_0 + 2 \cdot d_{max} - 2 \cdot l_{max}$ such that it finishes being sent by $t_0 + 2 \cdot d_{max} - l_{max}$ (i.e., within d_{max} from the activation time of node i). This packet of node i will only be able to reach its receiver, if node j is not activated anew until time $t_0 + 2 \cdot d_{max} - l_{max}$. Since the minimum overlapping between any two packets yields packet loss, in order that node i's n-th packet is not interfered by node j, this latter should not be activated anew until $t_0 + 2 \cdot d_{max}$ – see illustration in Fig. 4.2. The lemma follows. □

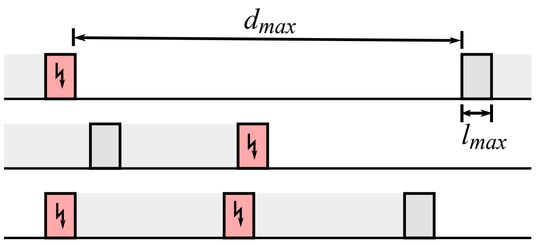


Figure 4.2.: Illustration of Lemma 1 for the case of three nodes. The last packet of the top node interferes with the first packet of the bottom node. Given that the second packet of the bottom node might also be lost, its third (and last) packet can only be guaranteed to reach its receiver, if the top node is forced to wait for at least d_{max} time before transmitting anew.

As a result of the above lemma, each node needs to transmit a minimum of n packets in order that at least one reaches the receiver within d_{max}, i.e., $k_i = n$ for $1 \leq i \leq n$. In addition, each node should only be activated once with $t_{act} \geq 2 \cdot d_{max}$, meaning that there is a transmission pause of at least d_{max} after each sequence.

Lemma 1 only considers the first packet being sent by each node in the system. However, each node sends a sequence of n packets. Any of these subsequent packets may collide with other packets of the different nodes leading to interference. As a result, Lemma 1 states necessary, but not sufficient conditions for a reliable communication. In other words, to guarantee that at least one packet reaches the receiver in the worst case, we need to perform a more detailed analysis.

4.1.2. Deriving inter-packet times

In principle, from the above discussion, we know that a packet from one node can be interfered by a packet of another node sending at the same time. In addition, the minimum overlapping between any two packets leads to packet loss, since data can be corrupted. The following theorem states a necessary and sufficient condition for guaranteeing that, among the n packets sent by a node, at least one of them reaches its receiver.

Theorem 1. *Let us consider a set of n independent transmit-only nodes. Each of them sends a sequence of n packets within d_{max} and is activated at most once within $t_{act} = 2 \cdot d_{max}$. At least one packet of each node can be guaranteed to reach its receiver, if the following condition holds for $1 \leq i \leq n$, $1 \leq j \leq n$, $1 \leq \alpha_i \leq n-1$, and $i \neq j$:*

$$\mathrm{mod}\left(\frac{\alpha_i \cdot t_i}{t_j}\right) \geq 2 \cdot l_{max}, \tag{4.1.1}$$

where $\mathrm{mod}(\cdot)$ is the modulo operation, l_{max} is the maximum length of a packet, while t_i and t_j are the (constant) inter-packet times of the i-th and the j-th node respectively.

The proof of Theorem 1 can be found in Appendix A.1. Now, in order to facilitate understanding and to visualize the above theorem, let us consider the following corollary.

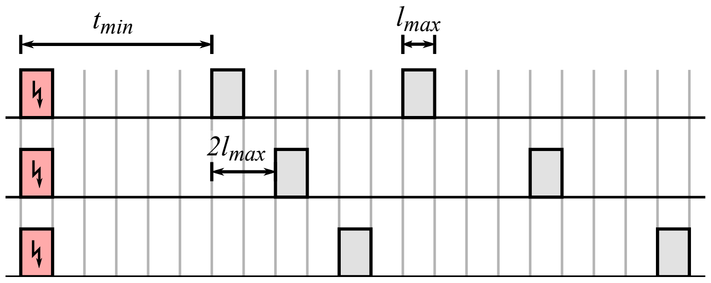


Figure 4.3.: Illustration of Corollary 1 for $n = 3$. In the case of a simultaneous activation, if (4.1.1) holds, the first packet of each node will get lost. However, there will be no further packet losses for the next two packets (i.e., $n - 1$) of each node.

Corollary 1. *Let us assume that, according to Theorem 1,* (4.1.1) *holds for* $1 \leq i \leq n$, $1 \leq j \leq n$, $1 \leq \alpha_i \leq n-1$, *and* $i \neq j$. *If all* n *transmit-only nodes are activated simultaneously at time* t_0, *the first packet of each such nodes will be lost; however, there will be no more packet losses in* $[t_0, t_0 + (n-1) \cdot t_i]$ *for* $1 \leq i \leq n$.

Proof. Let us consider that any pair of nodes i and j where $i \neq j$ are activated together at time t_0. If (4.1.1) holds for $1 \leq \alpha_i \leq n-1$, there is at least $2 \cdot l_{max}$ space between any packets of i and j in $[t_i, (n-1) \cdot t_i]$. This means that, with exception of the first packets sent at t_0, there is no overlapping between packets of node i and j in $[t_0, (n-1) \cdot t_i]$ – see illustration in Fig. 4.3. Further, if (4.1.1) holds for all i and j where $i \neq j$, no packets are lost in $[t_0, (n-1) \cdot t_i]$ and $1 \leq i \leq n$ besides the first one sent at t_0 by each node. The corollary follows. □

Theorem 1 allows us to guarantee that at least one packet of each node reaches its corresponding receiver within d_{max}, provided that each node send n packets within d_{max}. However, it does not help select the values of t_i for each of the nodes. To this end, let us consider the following analysis.

Lemma 2. *Let us consider a set of* n *independent transmit-only nodes. Each of them sends a sequence of* n *packets within* d_{max} *and is activated at most once within* $t_{act} = 2 \cdot d_{max}$. *In order to guarantee that at least one packet of each such nodes reaches its corresponding receiver, the following condition must hold for any* t_i *and* t_{i-1} *where* $1 < i \leq n$:

$$t_i - t_{i-1} \geq 2 \cdot l_{max}. \tag{4.1.2}$$

given that $t_{i-1} < t_i < 2 \cdot t_{i-1}$ *holds, i.e.,* $\lfloor \frac{t_i}{t_{i-1}} \rfloor = 1$.

Proof. According to Theorem 1, if (4.1.1) holds for all i and j where $i \neq j$ and $1 \leq \alpha_i \leq n-1$, it can be guaranteed that at least one packet of each node reaches its corresponding receiver.

Now, for a any i, $j = i-1$, and $\alpha_i = 1$, from (4.1.1) we have $\bmod\left(\frac{t_i}{t_{i-1}}\right) \geq 2 \cdot l_{max}$, which again has to hold according to Theorem 1. Since $t_i > t_{i-1}$ and $\lfloor \frac{t_i}{t_{i-1}} \rfloor = 1$ hold for $1 < i \leq n$, we have that $t_i - t_{i-1} \geq 2 \cdot l_{max}$ and the lemma follows. □

Theorem 2. *Let us consider a set of* n *independent transmit-only nodes. Each of them sends a sequence of* n *packets within* d_{max} *and is activated at most once within* $t_{act} = 2 \cdot d_{max}$. *If one packet of a node* i *is interfered by a packet of node* j, *in order to guarantee that the next packet sent by node* i *is not interfered again by node* j, *the following condition must hold for the minimum inter-packet time* $t_{min} = \min\limits_{1 \leq i \leq n}(t_i)$:

$$t_{min} \geq 2 \cdot n \cdot l_{max}, \tag{4.1.3}$$

where $t_{min} < t_i < 2 \cdot t_{min}$, *i.e.,* $\lfloor \frac{t_i}{t_{min}} \rfloor = 1$, *for* $1 \leq i \leq n$.

Proof. Let us assume that one packet of a node i is interfered at time t_0 by a packet of node j. In order that the next packet sent by node i is not interfered again by node j, according to Theorem 1, (4.1.1) needs to hold for all i and j where $i \neq j$ and $\alpha_i = 1$.

Without loss of generality, let us assume that all t_i are sorted in order of increasing values, i.e., $t_i > t_j$ if $i > j$ (hence $t_{min} = t_1$ holds). Since $\lfloor \frac{t_i}{t_{min}} \rfloor = 1$ is assumed to hold, note that $\lfloor \frac{t_i}{t_j} \rfloor = 1$ also holds for all i and j.

Now, if $i = n$, $j = n-1$, and $\alpha_n = 1$ hold, from (4.1.1) we have $\mod\left(\frac{t_n}{t_{n-1}}\right) = t_n - t_{n-1} \geq 2 \cdot l_{max}$, which leads to $t_n \geq t_{n-1} + 2 \cdot l_{max}$. Similarly, if $i = n-1$, $j = n-2$, and $\alpha_{n-1} = 1$, we obtain $t_{n-1} \geq t_{n-2} + 2 \cdot l_{max}$. Proceeding as before for $1 < i \leq n$ and $1 \leq j \leq i$, we obtain that $t_2 \geq t_1 + 2 \cdot l_{max}$ and, hence, $t_n \geq t_1 + 2 \cdot (n-1) \cdot l_{max}$.

Since $t_{min} = t_1$, $t_1 < t_n < 2 \cdot t_1$ holds by assumption. In addition, (4.1.1) is assumed to hold for all i and j where $i \neq j$ and $1 \leq \alpha_i \leq n-1$. Hence, for $i = 1$, $j = n$, and $\alpha_1 = 2$, (4.1.1) becomes $\mod\left(\frac{2 \cdot t_1}{t_n}\right) = 2 \cdot t_1 - t_n = 2 \cdot t_1 - t_1 - 2 \cdot (n-1) \cdot l_{max} \geq 2 \cdot l_{max}$, which leads to $t_1 \geq 2 \cdot n \cdot l_{max}$. The theorem follows. □

Theorem 2 provides a lower bound for t_{min} for the case that $t_{min} < t_i < 2 \cdot t_{min}$ where $1 \leq i \leq n$, i.e., all t_i have similar values. This is a meaningful choice for t_i since all nodes have the same deadline d_{max}. Note that inter-packet times t_i that strongly differ from each other make it difficult to meet d_{max} with all nodes.

Theorem 3. *Let us consider a set of n independent transmit-only nodes. Each of them sends a sequence of n packets within d_{max} and is activated at most once within $t_{act} = 2 \cdot d_{max}$. If the first packet of a node i is interfered by a packet of node j, in order to guarantee that the next $(n-1)$ packets sent by node i are not interfered again by node j, the following condition must hold for the minimum inter-packet time $t_{min} = \min\limits_{1 \leq i \leq n}(t_i)$:*

$$t_{min} \geq 2 \cdot (n-2) \cdot (n-1) \cdot l_{max} + 2 \cdot l_{max}, \tag{4.1.4}$$

where as before $t_{min} < t_i < 2 \cdot t_{min}$ holds, i.e., $\lfloor \frac{t_i}{t_{min}} \rfloor = 1$, for $1 \leq i \leq n$. In addition, $\lfloor \frac{\alpha \cdot t_{min}}{(\alpha-1) \cdot t_{max}} \rfloor = 1$ also holds, i.e., $(\alpha - 1) \cdot t_{min} < (\alpha - 1) \cdot t_{max} < \alpha \cdot t_{min}$, for $1 < \alpha \leq n-1$ and $t_{max} = \max\limits_{1 \leq i \leq n}(t_i)$.

Proof. Let us again assume that the first packet sent by a node i is interfered at time t_0 by a packet of node j. In order that the next $(n-1)$ packets sent by node i are not interfered again by node j, (4.1.1) needs to hold for all i and j where $i \neq j$ and $1 \leq \alpha_i \leq n-1$ as per Theorem 1.

Without loss of generality, let us assume that all t_i are sorted in order of increasing values, i.e., $t_i > t_j$ if $i > j$. Hence $t_{min} = \min\limits_{1 \leq i \leq n}(t_i) = t_1$ and $t_{max} = \max\limits_{1 \leq i \leq n}(t_i) = t_n$ hold.

Let us first consider $i = 1$ and $j = n$. If $\alpha_i = 1$ holds, from (4.1.1) we have that $\mod\left(\frac{t_1}{t_n}\right) \geq 2 \cdot l_{max}$ is equal to $t_1 \geq 2 \cdot l_{max}$ since $t_1 < t_n$. For $\alpha_i = 2$, from (4.1.1) we have that $\mod\left(\frac{2 \cdot t_1}{t_n}\right) \geq 2 \cdot l_{max}$ is equal to $2 \cdot t_1 - t_n = t_1 - 2 \cdot (n-1) \cdot l_{max}$, as $\lfloor \frac{2 \cdot t_1}{t_n} \rfloor = 1$ holds – see again proof of Theorem 2. Similarly, for $\alpha_i = 3$, we have that $\mod\left(\frac{3 \cdot t_1}{t_n}\right) \geq 2 \cdot l_{max}$ is equal to $3 \cdot t_1 - 2 \cdot t_n = t_1 - 2 \cdot 2 \cdot (n-1) \cdot l_{max} \geq 2 \cdot l_{max}$, as $\lfloor \frac{3 \cdot t_1}{2 \cdot t_n} \rfloor = 1$ holds. For $\alpha_i = n-1$, we have that $\mod\left(\frac{(n-1) \cdot t_1}{t_n}\right) \geq 2 \cdot l_{max}$ is equal to $(n-1) \cdot t_1 - (n-2) \cdot t_n = t_1 - (n-2) \cdot 2 \cdot (n-1) \cdot l_{max} \geq 2 \cdot l_{max}$, as $\lfloor \frac{(n-1) \cdot t_1}{(n-2) \cdot t_n} \rfloor = 1$ also holds. As a result, we have that $t_1 \geq 2 \cdot (n-2) \cdot (n-1) \cdot l_{max} + 2 \cdot l_{max}$ which is lower bound for $t_{min} = t_1$ stated in (4.1.4). Since $t_n = t_{max}$, note that choosing another j where $1 < j < n$ yields a lower bound that is closer to that of Theorem 2. In other words, the lower bound of (4.1.4) is the greatest necessary value of t_{min}. The theorem follows. □

Similar to Theorem 2, Theorem 3 provides a lower bound on t_{min} for the case that $t_{min} < t_i < 2 \cdot t_{min}$ where $1 \leq i \leq n$. However, in contrast to Theorem 2, the lower bound of Theorem 3 guarantees that, if a packet of node i gets interfered by any node j, its next $n-1$ packets will not be interfered again by node j. This result, together with Corollary 1, allows us to design a reliable communication network based on transmit-only nodes, since we can guarantee that at least one packet of each node reaches its receiver in the worst case.

Once we have found t_{min} by using directly Theorem 3, we can use Lemma 2 to obtain each t_i. Note that this choice of t_i allows meeting d_{max}, if and only if the following corollary holds for it.

Corollary 2. *Let us consider a set of n independent transmit-only nodes. Each of them will be able to send a sequence of n packets within d_{max} if the following condition holds for $1 \leq i \leq n$:*

$$d_{max} \geq (n-1) \cdot t_i + l_{max}, \tag{4.1.5}$$

where t_i is the (constant) inter-packet time of the i-th node.

Proof. A node i sends n packets with constant inter-packet time t_i. If it starts sending its first packet at time t_0, it will send its subsequent $n-1$ packets at times $t_0 + \alpha_i \cdot t_i$ where $1 \leq \alpha_i \leq n-1$. Node i finishes sending its n-th packet at time $t_0 + (n-1) \cdot t_i + l_{max}$. The corollary follows. □

In conclusion, this section presented an analytic model to calculate the missing MAC parameters k and t, i.e., packet numbers and inter-packet times. Compared to other, heuristic approaches such as [10] or [11], the presented analytical model strongly reduces computational complexity and, therefore, allows assessing parameters and performance in much shorter time — this is analyzed in Section 6.3.1. However, it is based on the assumption that nodes are the same and, hence, results in lower performance if adapted for heterogeneous networks. To remove this restriction and improve performance in such networks, a heuristic model is presented next.

4.2. Heuristic model

In the previous chapter, an analytic model was derived that is based on the assumption that deadlines and packet lengths are the same for all nodes in the network. Although this is the case in many applications, there are also numerous scenarios which consist of many different node types. The analytical model can be adapted to such networks, but typically results in lower performance, since the longest packet length and shortest deadline have to be assumed for all nodes. This section removes this limitation and presents a heuristic model that is explicitly designed for heterogeneous networks. To this end, this section first determines the packet numbers k, then derives the conditions and boundaries of inter-packet times t_i, and, finally, presents a heuristic search algorithm to find safe values for t_i.

4.2.1. Selecting packet numbers

Since the presented heuristic model allows for arbitrary deadlines d_i and packet lengths l_i, the existing formulas of the analytic model in Section 4.1 cannot be used anymore, but these have to be derived again. In the following, Lemma 3 first states the necessary requirements and conditions for the packet numbers k_i for different cases.

Lemma 3. *Let us consider a set of n independent transmit-only nodes. Each node j, with $1 \leq j \leq n$, is activated once and sends a sequence of k_j packets with constant inter-packet times t_j within its deadline d_j. Further, we consider that t_i and t_j can be chosen such that* *there is at most one collision between any two sequences of any two nodes j and i. In the worst case, at most $\gamma_i - 1$ packets of node i, with $1 \leq i \leq n$ and $i \neq j$, will be lost due to starting sequences of every node j:*

$$\gamma_i = n + \sum_{j=1, j \neq i}^{n} \left\lfloor \frac{d_i}{d_j} \right\rfloor . \tag{4.2.1}$$

Proof. Since nodes are independent of each other, they start transmitting their sequences of packets at arbitrary points in time. Hence, it may happen that every time a node i transmits a packet, this gets interfered by a packet of a sequence of another node j — with $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$ — being sent at that time. Since there are $n-1$ nodes other than i in the system, $n-1$ packets of node i can be interrupted this way, provided that suitable t_i and t_j can be found such that there is at most one collision between any two sequences of nodes j and i.

In addition, since each of the other $n-1$ nodes is activated at most once within its deadline d_j, node i's transmissions can be interfered at most $\lfloor \frac{d_i}{d_j} \rfloor$ times more — in total $\lfloor \frac{d_i}{d_j} \rfloor + 1$ times — by the same node j. As a result, when considering all nodes in the system, the following expression can be obtained:

$$\sum_{j=1, j \neq i}^{n} \left(\left\lfloor \frac{d_i}{d_j} \right\rfloor + 1 \right) = n - 1 + \sum_{j=1, j \neq i}^{n} \left\lfloor \frac{d_i}{d_j} \right\rfloor ,$$

which is $\gamma_i - 1$ as defined in (4.2.1). The lemma follows. □

Lemma 3 states that a node i needs to transmit a minimum of γ_i packets in order that at least one of them reaches its destination within d_i in the worst case, i.e., this is a *safe* value for k_i. However, since (4.2.1) is based on the fact that nodes can be activated at arbitrary points in time, it may result in very pessimistic values. Note that Lemma 3 considers starting sequences of packets by a node j. As discussed later, collisions with subsequent packets in a node j's sequence can be prevented by selecting suitable values of t_i and t_j.

Let us consider the following example consisting of two nodes with $d_i = 1\,\text{min}$ and $d_j = 500\,\text{ms}$ respectively. From (4.2.1), we obtain $\gamma_i = 121$, i.e., in spite of having a two-node network, node i needs to sends at least 121 packets to achieve reliability in the worst case. If we now have another node with a 500 ms deadline, γ_i will further increase by 120, i.e., node i will have to send 241 packets.

To countervail this pessimism, it is possible to impose a transmission pause of at least d_{max} after each sequence of packets, where d_{max} denotes the maximum deadline in the network. This is formalized in the next lemma.

Lemma 4. *Let us consider a set of n independent transmit-only nodes. Each node j sends a sequence of k_j packets with constant inter-packet times t_j within its deadline d_j, after which it implements a transmission pause of at least d_{max}. Further, we consider that t_i and t_j can be chosen such that there is at most one collision between any two sequences of any two nodes j and i. In the worst case, at most $n-1$ packets of node i will be lost due to starting sequences of packets of every node j, where $d_{max} = \max_{j=1}^{n}\{d_j\}$, $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$ hold.*

Proof. Since nodes are independent of each other, they start transmitting their sequences of packets at arbitrary points in time. As already mentioned, every time a node i transmits a packet, this can be interfered by a packet of a sequence of another node j — with $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$ — being sent at that time. Since there are $n-1$ nodes other than i in the system, $n-1$ packets of node i can be interrupted this way, provided that suitable t_i and t_j can be found such that there is at most one collision between any two sequences of nodes j and i.

Now, since each of the other $n-1$ nodes can be activated not earlier than d_{max} time after having finished one sequence of packets, node i's transmissions cannot be interfered anew by a starting sequence of packets of any other node j, i.e., $\lfloor \frac{d_i}{d_{max}+l_j} \rfloor = 0$. As a result, if transmission pauses greater than or equal to d_{max} are enforced after each sequence of packets, in the worst case, at most $n-1$ packets can be lost by any node i. The lemma follows. □

Unfortunately, in contrast to the approaches from the literature, imposing a transmission pause of at least d_{max} after each sequence of packets is not a suitable solution due to considering arbitrary deadlines. In the above example, this would lead to node j being *blocked* by 1 min after each activation. If node j is a light switch and node i a temperature sensor in a home-automation setting, this means that lights would be blocked in an on- or off-state for 1 min, which is clearly an unacceptable delay.

To reduce pessimism by (4.2.1) without enforcing long transmission pauses, the concept of delayed activation is introduced as illustrated in Fig. 4.4. In principle, after every sequence of packets of any node j, an event E is not allowed to trigger a new sequence of packets immediately. This is rather delayed to the closest time instant that is a multiple of t_j starting from the last packet of the previous sequence.

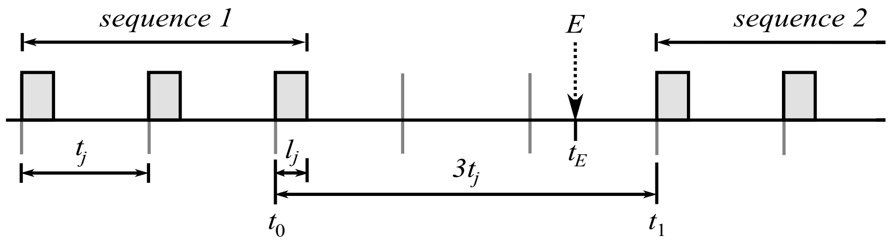


Figure 4.4.: Delayed-activation scheme: An event E is not allowed to immediately trigger a new sequence of packets of a node j. Instead, this is delayed to the next time instant that is a multiple of t_j starting from the last packet of the previously sent node j's sequence.

That is, if the last packet of node j's *sequence 1* is sent at time t_0 and an event E occurs at a later time t_E, then the next sequence of packets (triggered by E) starts at a t_1 being $t_E \leq t_1$ where $t_1 - t_0$ is an integer multiple of t_j — in Fig. 4.4 this is $3t_j$.

Lemma 5. *Let us consider a set of n independent transmit-only nodes. Each node j sends a sequence of k_j packets with constant inter-packet times t_j within its deadline d_j, after which it implements a delayed activation as described above for a time interval of length d_{max}. In the worst case, at most $n-1$ packets of a node i will be lost due to starting sequences of packets of every node j provided that suitable t_i and t_j can be found, where $d_{max} = \max_{j=1}^{n}\{d_j\}$, $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$ hold.*

Proof. Let us consider again the example of Fig. 4.4. In the case $t_E - t_0 - l_j > d_{max}$, if node j's *sequence 1* interfered with a packet of a sequence of node i, node j's *sequence 2* triggered by E cannot interfere with any other packet of the same node i's sequence as per Lemma 4.

In the case $t_E - t_0 - l_j < d_{max}$, if node j's *sequence 1* interfered with a packet of a sequence of node i, by properly selecting t_j and t_i, it can be avoided that *sequence 2* and its following node j's sequences in $[t_0 + l_j, t_0 + l_j + d_{max}]$ interfere with any other packet of same node i's sequence. Note that if proper t_i and t_j can be found for any i and j with $i \neq j$, in the worst case, at most $n-1$ packets of any node i can be lost due to starting sequences of every other node j. The lemma follows. □

As per Lemma 5, the delayed-activation scheme allows us to reduce the pessimism introduced by Lemma 3 in the same way Lemma 4 does, but without enforcing long transmission pauses. For this, suitable values of t_i and t_j must be configured for every i and j where $i \neq j$ as discussed in the next section.

To further clarify this, let us again consider an exemplary system composed of 9 light switches with $d_{max} = 500\,\mathrm{ms}$ and one temperature sensor with $d_{max} = 1\,\mathrm{min}$. If we implement a transmission pause after each sequence, like in [10] or in the analytic model in Section 4.1, each light switch will have a pause time of $1\,min$ after each activation, which is clearly unacceptable. In case of the delayed activation scheme, this pause time reduces to roughly $\frac{500\,\mathrm{ms}}{10} = 50\,\mathrm{ms}$ for the light switch (and $6\,\mathrm{s}$ for the temperature sensor). Now, if the user accidentally switches a light on and wants to switch it immediately back off, he must wait at most $500\,\mathrm{ms}$ (instead of $1\,\mathrm{min} + 500\,\mathrm{ms}$), since a node can be activated at most once within its deadline. As a result, the delay incurred by our delay-activation scheme is small and can typically be neglected.

4.2.2. Inter-packet time conditions and boundaries

So far, collisions have been caused by the first packet sent by other nodes in the network. However, any of their subsequent packets may also produce collisions leading to further packet loss. In other words, sending sequences of n packets in the delayed-activation scheme, i.e., making $k_i = n$ for $1 \leq i \leq n$, is necessary but not sufficient to guarantee a reliable communication.

To guarantee full reliability, i.e., that at least one packet of a node reaches its receiver in the worst case, *safe* values of t_i must also be selected for each node in the system. Towards this, note that a packet from one node can be interfered by a packet of another node sending simultaneously, and that the minimum overlapping between any two packets leads to packet loss. The following theorem states a necessary and sufficient condition for guaranteeing that, among the n packets sent by a node i, at least one of them reaches its destination.

Theorem 4. *Let us consider a set of n independent transmit-only nodes, each of which is activated once and sends a sequence of $k_i = n$ packets with constant inter-packet times t_i within its deadline d_i. In the worst case, at least one packet of each node can be guaranteed to reach its destination, if the delayed-activation scheme is used and the following condition holds for $1 \leq i \leq n$, $1 \leq j \leq n$, $i \neq j$, and $1 \leq \alpha_i \leq n-1$:*

$$\mathrm{mod}\left(\frac{\alpha_i \cdot t_i}{t_j}\right) \geq l_i + l_j, \tag{4.2.2}$$

where $\mathrm{mod}\,(\cdot)$ is the modulo operation, l_i and l_j are the packet lengths of node i and j respectively, while t_i and t_j are their corresponding inter-packet times.

The proof of Theorem 4 can be found in A.2. This ensures that at least one packet of each node reaches its destination within d_i, provided that each node follows the delayed-activation scheme and sends n packets within d_i. It does not help in selecting suitable values of t_i, but only verifies provided values of t_i.

The following is about finding *safe* values for t_i for any i and $1 \leq i \leq n$. However, deriving t_i analytically is difficult, hence, an algorithm which heuristically searches for valid values of t_i is proposed instead. To this end, the following lemma provides an upper bound on the values of t_i.

Lemma 6. *If a node i follows the delayed-activation scheme and sends n packets within d_i according to Lemma 5, then its (constant) inter-packet time is upper bounded by $\hat{t}_i$, where l_i is node i's packet length:*

$$\hat{t}_i = \frac{d_i - l_i}{n}. \tag{4.2.3}$$

Algorithm 1 Searching for values of t_i

```
Require: $n$, list of $(l_i, d_i)$
 1: sort list of $(l_i, d_i)$ according to non-decreasing $d_i$
 2: for $i = 1$ to $n$ do
 3:    found = false
 4:    $t_{temp} = \frac{d_i - l_i}{n}$
 5:    while $t_{temp} > 0$ do
 6:       if check_period $(i, t_{temp})$ == true then
 7:          found = true
 8:          $t_i = t_{temp}$
 9:          break
10:       else
11:          $t_{temp} = t_{temp} - step$
12:       end if
13:    end while
14:    if found == false then
15:       return (failed)
16:    end if
17: end for
18: return (list of $t_i$)
```

Proof. Without loss of generality, let us assume that node i is activated at time t_0. In order that n packets can be sent within $[t_0, t_0 + d_i]$, the n-th packet has to be sent at latest at $t_0 + d_i - l_i$. This way, node i's n-th packet finishes being sent exactly at $t_0 + d_i$.

On the other hand, according to the delayed-activation scheme, the transmission of a sequence of packets can be delayed by at most t_i — see again Fig. 4.4. Hence, the first node i's packet will be sent at $t_0 + t_i$. Since t_i is assumed to be constant, it should fit $n-1$ times in an interval of length $d_i - l_i - t_i$ in the worst case. Now replacing t_i by $\hat{t}_i$ to denote the greatest possible t_i, this leads to $\hat{t}_i = \frac{d_i - l_i - \hat{t}_i}{n-1}$, which results in $\hat{t}_i = \frac{d_i - l_i}{n}$. The lemma follows. □

4.2.3. Inter-packet time search algorithm

Alg. 1 computes values of t_i for a set of n transmit-only nodes, given packet lengths l_i and deadlines d_i with $1 \leq i \leq n$. The algorithm iterates over a list of ordered pairs (l_i, d_i) — see line 2, which has been sorted according to non-decreasing d_i. For each i, Lemma 6 is applied at line 4 to compute the longest possible t_i, temporally stored in t_{temp}, that allows meeting the deadline d_i. This way, the algorithm intends to maximize t_i. The greater the value of t_i, the less packets will be sent per time unit leading to less collisions.

This t_{temp} is checked by function *check_period()* as discussed later. If this check is successful, the value of t_{temp} is adopted for the current i — see line 8. If the check fails, t_{temp} is decremented in line 9 by a amount equal to *step*[2] as long as it remains greater than zero — see while-loop at line 5. If t_{temp} becomes zero, it means that *check_period()* was never successful and the algorithm returns *failed* at line 15. Otherwise, if the algorithm finds valid values of t_i for every i, these are returned as a list.

Alg. 2 shows the function *check_period()*, which is invoked from Alg. 1 and verifies a given value of t_{temp}. The algorithm is based on Theorem 4, i.e., it iteratively computes (4.2.2). If (4.2.2) holds for t_{temp} and all t_j for $j \leq i - 1$ in lines 9 to 15, i.e., the previously computed inter-packet times, t_{temp} is a valid value for t_i and *check_period()* returns *true* — otherwise it returns *false*. For this, *check_period()* requires the list of ordered pairs (l_j, t_j), i.e., the inter-packet times obtained so far, as well as index i of the node whose t_i is currently being

[2]The smallest meaningful value of *step* is equal to the time required for transmitting one bit on the communication channel.

Algorithm 2 Function *check_period()*

```
Require: $n$, $i$, $t_{temp}$, list of $(l_j, t_j)$ for $j < i$
 1: for $j = 1$ to $i - 1$ do
 2:     if $t_j < t_{temp}$ then
 3:         $t_{long} = t_{temp}$
 4:         $t_{short} = t_j$
 5:     else
 6:         $t_{long} = t_j$
 7:         $t_{short} = t_{temp}$
 8:     end if
 9:     for $k = 1$ to $n - 1$ do
10:         if mod $\left(\frac{k \cdot t_{long}}{t_{short}}\right) < l_{long} + l_{short}$ then
11:             return (false)
12:         else if $t_{short} -$ mod $\left(\frac{k \cdot t_{long}}{t_{short}}\right) < l_{long} + l_{short}$ then
13:             return (false)
14:         end if
15:     end for
16: end for
17: return (true)
```

computed. To simplify the computation of (4.2.2), t_{temp} and t_j are compared and stored in t_{long} and t_{short} depending on their values — see lines 2 to 8 in Alg. 2.

Note that, according to Theorem 4, (4.2.2) needs to be computed once with t_{long} and once with t_{short} in the numerator. The case with t_{long} in the numerator is checked in line 10, whereas the case with t_{short} in the numerator is checked in line 12. This way, it is possible to reduce the number of iterations that would be otherwise necessary.

4.3. Extension to arbitrary packet numbers

So far, each node in the system sends n packets leading to fully reliable communication within the network, i.e., without external interference. In some applications, however, it is favorable to trade reliability for a reduced number of packets in order to save energy. For example, pressing a light switch in a home-automation setting is not safety critical and the user can press it again, if the light does not turn on/off. Clearly, the reliability should still be high enough not to negatively impact the system quality.

In this section, a method is presented to calculate the transmission reliability for each node i, if it sends a reduced number of $k_i \leq n$ packets. This allows adjusting packet numbers individually for each node to save energy whenever data loss can be tolerated. Note that, by changing the number of packets per sequence and keeping the periods found by Alg. 1, it is ensured that nodes with mixed reliability can coexist without affecting each other's performance. Further, this allows changing packet numbers dynamically depending on the message priority. For example, a node can use higher k for important alarm messages and a lower k for less important messages to save energy.

Now, it is assumed that all nodes in the network are activated and sending packets and, hence, produce interference. Since nodes send packets periodically, the probability of interfering with a packet of a node i at the communication channel is $\sigma_i = \frac{l_i}{t_i}$ [87], i.e., the node's duty cycle. However, by properly selecting inter-packet times, DEEP guarantees that, within one sequence of packets, there will be at most one interference with another node in the network.

Without loss of generality, it is assumed that nodes are sorted in order of decreasing σ_i, i.e., $\sigma_1 \geq \sigma_2$ and $\sigma_2 \geq \sigma_3$, etc. In other words, the smaller a node's index, the higher the probability of interfering with it. As a result, the greatest probability of losing $n-1$ packets is that of node n.

The following equation now defines $q_i(x)$, i.e., the probability that exactly x out of k_i packets sent by node i are lost due to internal interference. This results in:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j\neq i}^{n} \sigma_j \left(\sum_{l=1;\ l\neq i,j}^{n} \sigma_l \cdots \right) \right), \tag{4.3.1}$$

where $1 \leq x \leq (n-1)$ and $q_i(n) = 0\ \forall i$, i.e, the probability of losing all n packets is zero for every node i. Note that there are exactly x summations in the above equation.

A detailed derivation of (4.3.1) can be found in A.3. In particular, $\binom{k}{x}$ is the binomial coefficient and accounts for the different combinations of packets that may collide within a sequence. For example, if we want the probability of $x = 1$ packet out of $k = 3$ being interfered, there are $\binom{3}{1} = 3$ possibilities: Either the first, the second or the third packet may be interfered. Further, there can be at most one collision with any other node per sequence, hence, $q_i(x)$ does only depend on k_i and not on k_j of the other nodes j with $j \neq i$.

The remaining part of (4.3.1) considers the combinations of different nodes that can cause collisions within the sequence. For example, if there are 4 nodes, each packet of node 4 can be either corrupted by node 1, 2 or 3. In case multiple packets are lost ($x > 1$), different combinations of nodes can cause these losses. For example, if two packets are lost ($x = 2$), this can be because of collisions with node 1 and 2 or node 2 and 3, etc.

Finally, equation (4.3.1) can be simplified to (4.3.2) by assuming the same σ_i for all nodes, i.e., $\sigma_i = \sigma\ \forall i$:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\prod_{j=1}^{x} (n-j) \right) \cdot \sigma^x. \tag{4.3.2}$$

To illustrate the previous equations, let us calculate the loss rate of an exemplary network, i.e., the probability of loosing all k packets per sequence $q_i(k)$ with $1 \leq k \leq (n-1)$. To this end, n is set to four and the duty cycle of each node to $\sigma = 0.035$ — this is greatest duty cycle found by Alg. 1 for $n = 4$, $l_i = 187.5\,\mu s$, and $d_i = 500\,\text{ms}$. Using (4.3.2), loss rates are $q(4) = 0\,\%$, $q(3) \leq 0.026\,\%$, $q(2) \leq 0.74\,\%$ and $q(1) \leq 10.5\,\%$. It can be seen that a higher k results in a smaller $q(k)$. However, only relatively low k are required to achieve loss rates below $1\,\%$, e.g., only $k = 2$ in this case. This once again shows that full reliability, i.e., a loss rate of $q(n) = 0\,\%$, requires high costs in the form of increased packets numbers and delays. On the other hand, whenever data loss can be tolerated to some extent, the number of redundant packets can be reduced to save energy. For example, reducing k to 2 halves the energy consumption and only results in an average expected packet loss of less than $1\,\%$.

However, reducing the packet numbers has further consequences, for example, when external interference is present. This will be discussed in the following section.

4.4. External interference

External interference can occur, for example, when microwaves, wireless toys, etc. are turned on, or when there exist neighboring WSNs that have not been regarded during the design phase. Although this effect can be strongly reduced, for example, by selecting a more robust modulation scheme or a less-noisy frequency channel, as discussed previously in Section 2.2.2, it can not always be avoided and system performance can be jeopardized. For this reason, this section extends the previous theory to account for external interference.

The following analysis is based on the models and assumptions from Section 1.3, in which external interference is mathematically described by a *duty cycle* σ, i.e., the largest possible ratio of pulse width to inter-pulse separation of the noise. Note that for simplicity, it is assumed that σ is the same for all nodes in the network, i.e., these are all affected equally by interference.

This σ gives also the greatest probability of encountering an external interference pulse at the communication channel [87] — note that $\sigma \leq 1$ holds. A $\sigma = 1$ means that external interference occupies the full channel and, hence, no reliable communication is possible. This probability is clearly independent of $q(n)$ in (4.3.1), i.e., the probability of packet loss due to internal interference.

In general, data packets can be either lost by external or by internal interference. Considering an exemplary network with $n = 4$ nodes and $k = 2$ data packets per sequence, there are the following possible combinations that lead to full packet loss: (i) all packets are lost due to external interference, (ii) one packet is lost due to internal and one due to external interference and (iii) all packets are lost internally. Changing k to higher values results in more combinations of mixed interference in (ii). For example, $k = 3$ packets can be either lost by 1 or 2 internal collisions and 2 or 1 external ones. If $k = n$, there is no full packet loss due to internal collisions, since DEEP prevents this by construction.

Now, the previous example can be generalized to calculate $\hat{q}_i$, i.e., the probability to loose all k_i packets of a sequence of node i by internal and external interference. This results in:

$$\hat{q}_i = \sigma^{k_i} + \sum_{j=1}^{k_i} q_i(j) \cdot \sigma^{k_i - j} \cdot \bar{\sigma}^j, \tag{4.4.1}$$

where $1 \leq i \leq n$, $1 \leq k_i \leq n$ $\forall i$ and $q_i(j)$ is the probability of node i to loose j packets internally as per (4.3.1) or (4.3.2).

The first part of (4.4.1) considers the probability to lose all packets externally. The second part combines internal and external interference as well as just internal interference when $j = k_i$, i.e., the last term of the summation. Note that simulation-based experiments can be found in Section 6.4.4 that visualize the above equations. Next, the following section incorporates clock drifts into the model of DEEP.

4.5. Clock drift

All clocks used in electronic devices show a deviation in frequency with respect to each other, i.e., they *count* time at different rates. This deviation is known as clock drift and normally depends on a number of different factors such fabrication-induced variability, operating temperature, etc. As a result, since transmit-only nodes cannot be synchronized [14] [51], they will unavoidably have different time scales — see also Section 2.2.2. To account for these effects, this section extends DEEP to consider clock drift.

Since a node *counts* for t_i time before sending a packet, a clock drift leads to an *absolute* waiting time $\bar{t}_i$ different than t_i, i.e., the time without clock drift. As a result, this needs to be considered when selecting inter-packet times. In particular, (4.2.3) needs to be modified to guarantee that the node sends its n packets within d_i independent of clock drift. To this end, $\Delta\hat{t}_i$ denotes the maximum possible clock deviation (with respect to an ideal, non-drifting clock) in an interval of length $\hat{t}_i$. Analogously to Lemma 6, it is proceeded as follows to incorporate clock drift into (4.2.3):

$$\begin{aligned}(n-1)\cdot(\hat{t}_i+\Delta\hat{t}_i) &\leq d_i-(\hat{t}_i+\Delta\hat{t}_i)-l_i,\\ \hat{t}_i &\leq \frac{d_i-l_i-n\cdot\Delta\hat{t}_i}{n}.\end{aligned} \tag{4.5.1}$$

Similarly, (4.2.2) needs to be modified to consider clock drift as shown in the following:

$$\bmod\left(\frac{\alpha_i\cdot t_i}{t_j}\right)\geq l_i+l_j+\Delta\hat{t}_i+\Delta\hat{t}_j, \tag{4.5.2}$$

where again $\Delta\hat{t}_i$ and $\Delta\hat{t}_j$ denote the maximum possible clock deviation in an interval of length $\hat{t}_i$ and $\hat{t}_j$ respectively.

Note that (4.5.1) and (4.5.2) require knowing $\Delta\hat{t}_i$ and $\Delta\hat{t}_j$ (which depend on $\hat{t}_i$ and $\hat{t}_j$ that are unknown). However, it is known that clock deviation due to clock drift increases with the length of the considered time interval. Since $\hat{t}_i<\tilde{t}_i$ and $\hat{t}_j<\tilde{t}_j$ hold for $\tilde{t}_i=\frac{d_i}{n}$ and $\tilde{t}_j=\frac{d_j}{n}$, it can be stated that $\Delta\hat{t}_i<\Delta\tilde{t}_i$ and $\Delta\hat{t}_j<\Delta\tilde{t}_j$ also hold, where $\Delta\tilde{t}_i$ and $\Delta\tilde{t}_j$ are the maximum clock deviations in the intervals of length $\tilde{t}_i$ and $\tilde{t}_j$. As a result, to resolve the above dependency, $\Delta\hat{t}_i$ and $\Delta\hat{t}_j$ in (4.5.1) and (4.5.2) can be safely replaced by $\Delta\tilde{t}_i$ and $\Delta\tilde{t}_j$.

If clock drift is not considered, when selecting t_i for all nodes, it might happen that there are multiple collisions between any two different nodes in one sequence of packets, which results in a transmission reliability less than 100 % in the worst case. However, since DEEP transmits n redundant packets, it is generally robust against clock drift, as shown later in simulation in Section 6.4.5.

4.6. Key findings

This chapter presented DEEP, a deterministic protocol that enables fully reliable communication in low-cost, unidirectional networks. It consists in each node sending its data as sequences of redundant packets, whereby packet numbers and inter-packet times are carefully selected to generate collision-free transmission patterns. This way, it can be guaranteed that at least one packet always reaches its destination in the worst case when neglecting external interference.[3]

In order to calculate the missing MAC parameters, i.e., the packet numbers and inter-packet times, two different models were presented: An analytic and a heuristic approach. The analytic model is based on a set of equations and formulas that allow calculating performance and parameters at very low complexity in short time. However, it is limited to networks with only one node type, i.e., all nodes in the network must have the same deadline and packet lengths, which restricts possible applications areas. The heuristic approach, on the other hand, is based on a heuristic algorithm, which allows for arbitrary node types and generally offers a higher performance — however, at the cost of slightly increased computational complexity. In summary, both models enable reliable communication, each with its own advantages. For a detailed comparison, see also Section 6.3.

The presented models were further extended to consider practical factors such as external interference and clock drift, and it was evaluated how sending a reduced number of redundant packets per sequence affects a node's reliability and energy consumption. Results of a simulation in Section 6.4 show that decreasing the number of packets greatly reduces energy consumption, while reliability decreases only slightly, and DEEP is generally robust against external interference and clock drift.

In summary, DEEP brings reliability into low-cost, unidirectional networks, i.e., an architecture which is highly error-prone and in which many existing MAC protocols fail to provide

[3]Interference can be greatly reduced by locating a network indoors [68] or choosing a robust modulation scheme, etc., as discussed in Section 2.2.2. Nevertheless, a simulation-based experiment in Section 6.4.4 shows that DEEP is generally robust, i.e., less affected by external interference.

any QoS guarantees. This makes DEEP particularly interesting for applications that need to be both cost-efficient and reliable at the same time. For example, safety critical devices in a home automation setting, such a heating control or security devices, can now use unidirectional communication to reduce costs [14] [68]. Compared to similar approaches from the literature [10] [14] [95], DEEP offers a higher reliability and can accommodate more nodes in the network — more on this later in Section 6.3.

Despite the above advantages, DEEP also has some disadvantages. In particular, it suffers from an increased communication overhead, since full reliability requires a large number of redundant packets and long inter-packet times. More precisely, each node must send as many packets as there are nodes in the system (i.e., $k = n$), and inter-packet times increase quadratically with the network size (i.e., n^2) — see (4.1.4). As a result, DEEP is generally only suitable for smaller networks, for example, home-automation applications with $n < 50$ nodes [68]. For a complete overview of possible applications areas, see also Section 7.1.

To counteract the above-mentioned disadvantages and reduce packet numbers and inter-packet times, two further protocols are presented in this work. One is a bidirectional extension of DEEP, called bi-DEEP that implements carrier sensing and ACKs to improve the average performance — more on this can be found in Appendix B.2. The other protocol, called RARE, is designed for unidirectional networks and is discussed in the following chapter.

Chapter 5.

The RARE protocol

The Random Reliable Protocol (RARE) is designed to offer high reliability in low-cost, unidirectional networks. Unlike DEEP, which guarantees full reliability, i.e., data always reaches its destination in the worst case, RARE tolerates some data loss. In other words, it can be configured to a user-specified reliability of less than 100 %, i.e., to a certain probability that data reaches its destination within a deadline in the worst case.

Tolerating packet losses significantly reduces pessimism compared to DEEP's deterministic model and allows the use of a probabilistic model. This has the advantage that, on the one hand, it facilitates integration into existing systems, in particular, if these are of probabilistic nature [55].[1] On the other hand, this avoids the high costs of full reliability, i.e., the extra time and energy that need to be invested. This is a meaningful goal, since many applications tolerate some data loss, in particular, those that transmit their data periodically. For example, sensors transmitting heart rate, running speed, etc. to a health monitoring application, or climate sensors reporting data to a weather station. Losing some packets in this context does not negatively impact the system's performance, as long as a certain reliability is not undershot.

Before starting with the working principle of RARE, let us briefly recall the underlying models and assumptions, as discussed before in Section 1.3. That is, the network consists of n transmit-only nodes and multiple receive-only sinks that are connected in a single-hop (star-topology) fashion. Data transfers can either be triggered by events or occur periodically, and at least one packets must arrive with a certain probability p within a deadline d at the sink to ensure normal (error-free) operation. Further, it is assumed that packet loss originates from simultaneous transmissions, whereas external interference is neglected first and analyzed separately to facilitate understanding.

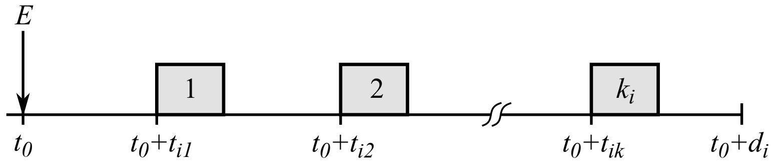


Figure 5.1.: Working principle of RARE: After being activated by an event E, node i transmits a sequence of k_i redundant packets in random time intervals within a deadline d_i.

The working principle of RARE is depicted in Fig. 5.1. Every time a node i is triggered by an event E, it transmits k_i redundant data packets of length l_i within a deadline d_i. In contrast to DEEP, inter-packet times are not constant, but node i waits a random time $t_{ix} \in \mathbb{R}_{>0}$ before sending any packet x — including the first one of a sequence. Here $1 \leq i \leq n$ and $1 \leq x \leq k_i$ hold and t_{ix} is uniformly selected from a time interval $[t_{min,i}, t_{max,i}]$ with $t_{min,i}, t_{max,i} \in \mathbb{R}_{>0}$. As a result, since each node waits a random time before every packet,

[1] Using a probabilistic approach facilitates the modeling of complex dependencies and random effects that are difficult to express otherwise. This results in a typically less complex mathematical model that is easier to handle and, therefore, easier to integrate.

transmissions are distributed equally over the available time (i.e., a node i's deadline d_i). This avoids peaks in traffic and balances network load, which reduces collision numbers, as shown later in simulation in Chapter 6. Finally, unlike DEEP, there is no transmission pause after a sequence and nodes can be triggered directly anew after their deadline has passed, i.e., $t_{act} = d_i$.

The RARE protocol has been previously published in [55] [67] [69]. More specifically, [69] contains the base model from Section 5.1, [67] the extended model with arbitrary node types from Section 5.2 and, in [55], RARE has been successfully integrated into an intelligent intersection management.

5.1. Deriving a base model

In this section, a mathematical model is derived to calculate the missing MAC parameters k_i, $t_{min,i}$ and $t_{max,i}$ to ensure a specific worst-case reliability p_i for communication between nodes in a network. To this end, a simplified system with only one node type is considered in order to reduce complexity and facilitate understanding. One node type means that all nodes in the network are the same, i.e., they all have the same deadline d_{max}, packet length l_{max}, packet numbers k, reliability p and inter-packet time boundaries t_{min} and t_{max}.[2] This is then extended later in Section 5.2 to consider arbitrary node types and practical factors such as external interference and clock drift. Again, although this assumption seems to be very restrictive, such networks are quite common. That is, many WSNs only monitor a few environmental conditions for which one node type is sufficient. For example, in a forest fire detection system [12], all nodes are equipped with a gas sensor to detect fire. By using the same nodes with the same hardware and software, etc., costs and complexity can be reduced.

Now, in order to guarantee a specific worst-case reliability p, suitable values for t_{min}, t_{max} and k have to be selected. Recall that reliability is defined as the probability that, in the worst case, at least one of a sequence of k packets of any node i reaches its destination within a specified deadline. To compute this probability, the worst-case transmission conditions need to be considered: (i) all n nodes in the network are sending packets, and (ii) every time a packet is sent by a node, there exists a maximum fraction of the interval $[t_{min}, t_{max}]$, denoted by Δ_{coll}, for which any selected value of t_{ix} leads to collisions. While condition (i) is straightforward, condition (ii) requires more analysis.

Recall that each node i in the network uniformly selects inter-packet times t_{ix} in $[t_{min}, t_{max}]$. Let us first consider the case where t_{min} is set such that there can be at most one packet of each node in an interval of length $t_{max} - t_{min}$. That is, the value of t_{min} has to fulfill the following condition:

$$\begin{aligned} t_{min} &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max}}{2}. \end{aligned}$$

Given this case, let us analyze the example of Fig. 5.2 for four nodes. Node 1 is activated at time t_0 and sends packets at $t_0 + t_{1,x}$ with $1 \leq x \leq k$. The probability of losing a packet is given by the probability of choosing a $t_{1,x}$ (from $[t_{min}, t_{max}]$) that leads to a collision with a packet of any of the other nodes. This probability is maximum, when the fraction of $[t_{min}, t_{max}]$ leading to potential collisions, i.e., Δ_{coll}, is the greatest possible.

In Fig. 5.2, there is a period of time in $[t_{min}, t_{max}]$ equal to $2l_{max}$, for which any $t_{1,x}$ leads to a collision with one of the other three nodes. This originates from the fact that even the

[2] In case there are different node types, these can be converted to one type by (pessimistically) assuming the shortest deadline and longest packet length for them, i.e., $d_{max} = \min_{i=1}^{n}\{d_i\}$ and $l_{max} = \max_{i=1}^{n}\{l_i\}$. The remaining parameters k, t_{min} and t_{max} will then be converted automatically.

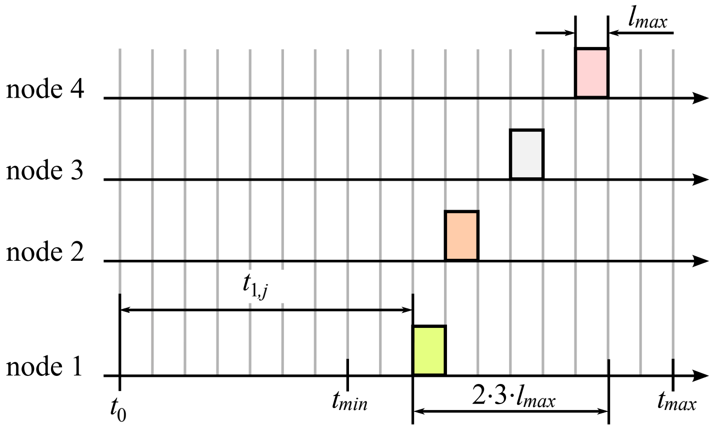


Figure 5.2.: Computing the worst-case probability of packet loss: This results from the ratio between the maximum number of inter-packet times that potentially yield a collision to the total number of possible inter-packet times.

smallest overlapping yields packet loss, i.e., a packet sent within $(t_{1,x} - l_{max}, t_{1,x} + l_{max}]$ will collide with packet x of node 1. In the worst case, the packets of the other three nodes are separated by at least a time equal to l_{max}. As a result, there will be three time intervals equal to $2l_{max}$ that lead to collisions. The sum of these time intervals results in the maximum value of Δ_{coll}, i.e., $2 \cdot 3 \cdot l_{max}$ in the example of Fig. 5.2. For n nodes, this leads to the following expression:

$$\Delta_{coll} = 2(n-1)l_{max}.$$

To generalize, the previous restriction is removed allowing t_{min} to have smaller values than $\frac{t_{max}}{2}$. Now, each node can send more than one packet within an interval of length $t_{max} - t_{min}$, for example, if it (randomly) selects t_{min} multiple times. We denote this number of packets by $m \in \mathbb{N}_{>0}$ for which t_{min} has to fulfill the following condition:

$$\begin{aligned} m \cdot t_{min} &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max}}{m+1}. \end{aligned} \tag{5.1.1}$$

Since there can be m packets of each node in an interval of length $t_{max} - t_{min}$, the generalized expression of Δ_{coll} is given by:

$$\Delta_{coll} = 2m(n-1)l_{max}. \tag{5.1.2}$$

As a consequence, the maximum possible probability of packet loss every time a packet is sent can be computed using the proposed MAC scheme by the ratio between Δ_{coll} and $t_{max} - t_{min}$:

$$q = \frac{2m(n-1)l_{max}}{t_{max} - t_{min}}. \tag{5.1.3}$$

Clearly, the probability of a successful packet transmission in the worst case is given by $1 - q$. Note that, for (5.1.3) to be valid the following condition must be satisfied (i.e., $q \leq 1$ must hold):

$$\begin{aligned} 2m(n-1)l_{max} &\leq t_{max} - t_{min}, \\ t_{min} &\leq t_{max} - 2m(n-1) \cdot l_{max}. \end{aligned} \tag{5.1.4}$$

In addition, since m, n, l_{max}, t_{max} and t_{min} are system parameters (i.e., common to all nodes in the WSN), q is independent of the node and of the packet being sent. As a result, it

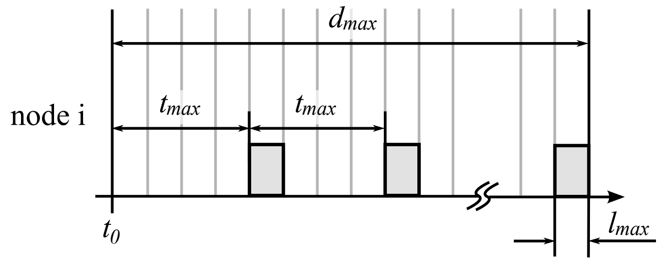


Figure 5.3.: Relation between d_{max} and t_{max}: In the worst case, t_{max} should fit k (number of packets sent) times in a time interval of length $d_{max} - l_{max}$.

is possible to model the transmission of packets in the network using a binomial distribution. This way, the probability p is computed that, in the worst case, at least one packet out of a sequence of k reaches its destination for any node in the network.

To compute p, all possible combinations need to be considered, i.e., only one packet arrives, two packets arrive, etc., which is a cumbersome procedure. It is much easier to compute the sequence loss rate $1-p$, i.e., the probability that, in the worst case, no packet of a sequence of k reaches its destination. This is the probability that k consecutive packets be lost and can be computed by the well-known equation $\binom{k}{z}q^z(1-q)^{(k-z)}$ where $\binom{k}{z} = \frac{k!}{z!(k-z)!}$ is the binomial coefficient. Replacing q as per (5.1.3) and selecting $z = k$, i.e., k out of k packets are lost, the binomial coefficient becomes 1, resulting in:

$$1 - p = \left(\frac{2m(n-1)l_{max}}{t_{max} - t_{min}}\right)^k. \tag{5.1.5}$$

In order that p corresponds to the previous definition of reliability, each nodes must send k packets within the specified deadline d_{max}. Towards this, recall again that every node i waits a random time t_{ix} chosen from $[t_{min}, t_{max}]$ before sending any packet. In the worst case, node i will have to wait t_{max} before sending each of its k packets as illustrated in Fig. 5.3. To guarantee that each node i sends its k packets within d_{max}, the following must hold:

$$t_{max} \leq \frac{d_{max} - l_{max}}{k}. \tag{5.1.6}$$

Given a value of t_{max} as per (5.1.6), (5.1.5) can be reshaped to compute the value of t_{min} that satisfies a desired reliability p for the whole WSN:

$$t_{min} \leq t_{max} - \frac{2m(n-1)l_{max}}{\sqrt[k]{1-p}}. \tag{5.1.7}$$

Note from (5.1.5) that a $p = 1$, i.e., 100% reliability, can only be achieved for $n = 1$, i.e., for only one node in the network, independent of all other parameters. For $n > 1$, if p tends to 1, t_{min} tends to minus infinity as per (5.1.7). In other words, 100% reliability as with the DEEP protocol cannot be achieved. However, RARE allows for a reliability that is acceptably close to 100%, while considerably reducing the number of packets sent and, hence, making better use of energy. A detailed simulation-based comparison can be found in Chapter 6.

5.2. Extension to arbitrary node types

The previous model assumed that all nodes within the network are the same, i.e., they have the same deadlines and packet lengths. Although such systems are common, there are also numerous scenarios which consist of many different node types, in particular, in IoT. The previous model can be adapted to such networks, but typically results in lower performance, since the longest packet length and shortest deadline have to be assumed for all nodes. This section removes these restrictions and adapts the previous theory to account for arbitrary packet lengths l_i, deadlines d_i, packet numbers k_i and reliabilities p_i for each node in the network.

To begin with, let us consider a node i with packet size l_i and a set of nodes j with packet sizes l_j that are sending packets simultaneously. The interval of time leading to collisions between node i and any node j is equal to $l_i + l_j$. Here again node i waits for a random time t_{ix} before sending a packet. However, t_{ix} is now uniformly distributed in $[t_{min,i}, t_{max,i}]$, where $t_{min,i}, t_{max,i} \in \mathbb{R}_{>0}$ are node-specific parameters. If every node j sends at most one packet in an interval of length $t_{max,i} - t_{min,i}$, this results in:

$$\Delta_{coll,i} = (n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j.$$

And, if every node j sends m packets in $t_{max,i} - t_{min,i}$, this changes to:

$$\Delta_{coll,i} = m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big), \tag{5.2.1}$$

where m has to fulfill the following condition for each node i and j in the network, i.e., at most m packets of node j can interfere with one packet of node i:

$$\begin{aligned} m \cdot t_{min,j} &\geq t_{max,i} - t_{min,i}, \\ t_{min,i} &\geq t_{max,i} - m \cdot t_{min,j}. \end{aligned} \tag{5.2.2}$$

As a consequence, node i's maximum possible probability of packet loss can be computed, which is given by the ratio between $\Delta_{coll,i}$ and $t_{max,i} - t_{min,i}$:

$$q_i = \frac{m\Big((n-1)l_i + \sum\limits_{j=1;j\neq i}^{n} l_j\Big)}{t_{max,i} - t_{min,i}}. \tag{5.2.3}$$

Clearly, the probability of a successful packet transmission by node i is $1 - q_i$ in the worst case. Note that, for (5.2.3) to be valid the following condition must be satisfied (i.e., $q_i \leq 1$ must hold):

$$t_{min,i} \leq t_{max,i} - m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big). \tag{5.2.4}$$

Note that q_i in (5.2.3) depends on node i; however, it is independent of the packet being sent by node i. As a consequence, the binomial distribution can be used again to compute the probability p_i that at least one out of a sequence of k_i node i's packets reaches its destination in the worst case. Proceeding as before, $1 - p_i$ is computed, i.e., the probability that none of the k_i packets of node i reaches its destination in the worst case:

$$1 - p_i = \left(\frac{m\Big((n-1)l_i + \sum\limits_{j=1;j\neq i}^{n} l_j\Big)}{t_{max,i} - t_{min,i}}\right)^{k_i}. \tag{5.2.5}$$

In order that p_i corresponds to this thesis' definition of reliability, it must be ensured that nodes always send k_i packets within their specified deadlines d_i:

$$t_{max,i} \leq \frac{d_i - l_i}{k_i}, \tag{5.2.6}$$

and, finally, solving (5.2.5) for $t_{min,i}$ results in:

$$t_{min,i} \leq t_{max,i} - \frac{m\Big((n-1)l_i + \sum\limits_{j=1;j\neq i}^{n} l_j\Big)}{\sqrt[k_i]{1-p_i}}. \tag{5.2.7}$$

In summary, the previous equations allow calculating safe values for $t_{min,i}$ and $t_{max,i}$ for each node i of a set of different nodes with arbitrary deadlines and packet sizes. Now, in order to assess the resulting impact on the system's reliability, the effect of arbitrary packet sizes is analyzed next. For this purpose, (5.2.4) is considered again, which gives an upper bound on $t_{min,i}$. This bound depends on $t_{max,i}$ and $\Delta_{coll,i}$ as per (5.2.1). Now, by increasing the packet sizes l_i, the collision interval increases and, hence, $t_{min,i}$ decreases. However, $t_{min,i}$ is lower bounded by (5.2.2) and can therefore only decrease to a certain extent. Otherwise, the system stops being feasible.

Reliability is then conditioned by the packet sizes l_i of all nodes as per (5.2.1) and by the length of the shortest $t_{min,j}$ in the system — as per (5.2.2). As a result, allowing for arbitrary packet sizes instead of assuming the longest possible l_{max} for every node reduces $\Delta_{coll,i}$ and, hence, increases reliability according to (5.2.5).

Now, before analyzing the effect of arbitrary deadlines on reliability, the following section further extends the presented theory to allow for arbitrary m.

5.2.1. Extension to different m

So far, the presented theory has considered that any node j can send at most m packets within an interval of length $t_{max,i} - t_{min,i}$, where m is the same for all nodes. According to (5.2.2), for any node i, this means that the shortest $t_{min,j}$ must fit no more than m times in $t_{max,i} - t_{min,i}$. As a consequence, nodes with long deadlines will be severely restricted by nodes with short deadlines and, hence, they cannot benefit from their long deadlines. To solve this problem, the case of different m for every node in the network is considered now:

$$m_{ij} = \left\lceil \frac{t_{max,i} - t_{min,i}}{t_{min,j}} \right\rceil, \tag{5.2.8}$$

where m_{ij} is the number of packets a node j can send in nodes i's interval $t_{max,i} - t_{min,i}$. As a result, (5.2.1) can be extended as follows:

$$\Delta_{coll,i} = l_i\Big(\sum_{j=1;j\neq i}^{n} m_{ij}\Big) + \sum_{j=1;j\neq i}^{n} m_{ij} \cdot l_j, \tag{5.2.9}$$

and (5.2.3) now becomes:

$$q_i = \frac{l_i\Big(\sum\limits_{j=1;j\neq i}^{n} m_{ij}\Big) + \sum\limits_{j=1;j\neq i}^{n} m_{ij} \cdot l_j}{t_{max,i} - t_{min,i}}. \tag{5.2.10}$$

Proceeding as before, reliability then changes from (5.2.5) to:

$$1 - p_i = \left(\frac{l_i\Big(\sum\limits_{j=1;j\neq i}^{n} m_{ij}\Big) + \sum\limits_{j=1;j\neq i}^{n} m_{ij} \cdot l_j}{t_{max,i} - t_{min,i}} \right)^{k_i}, \tag{5.2.11}$$

which can be reshaped to obtain $t_{min,i}$:

$$t_{min,i} \leq t_{max,i} - \frac{l_i\Big(\sum\limits_{j=1;j\neq i}^{n} m_{ij}\Big) + \sum\limits_{j=1;j\neq i}^{n} m_{ij} \cdot l_j}{\sqrt[k_i]{1-p_i}}. \tag{5.2.12}$$

Note that (5.2.6) reduces to (5.1.6) for $l_i = l_{max}$, $k_i = k$ and $d_i = d_{max}$. Similarly, (5.2.12) reduces to (5.1.7) for $t_{max,i} = t_{max}$, $l_i = l_j = l_{max}$, $k_i = k$ and $m_{ij} = m$. Here, p_i in (5.2.11) becomes p as per (5.1.5), i.e., the probability that at least one out of k packets reaches its destination in time is the same for all nodes in the network.

5.2.2. Optimizing reliability

This section discusses the effect of arbitrary deadlines on performance and presents an algorithm to calculate optimal m_{ij} and $t_{min,i}$ for improved reliability. As noted above, it is not meaningful to use the same m for every node, since a fixed m limits the benefits of modeling arbitrary deadlines in the network.

Similar to packet sizes l_i, deadlines d_i have an impact on nodes' reliability. Whereas different l_i have an influence on the collision interval as per (5.2.9), different d_i have an impact on $t_{max,i}$ as shown in (5.2.6). As a result, the length of the interval $t_{max,i} - t_{min,i}$ varies with d_i and, correspondingly, this influences p_i as per (5.2.11).

However, nodes will also have higher m_{ij} for greater deadline ratios as described in (5.2.8). That is, although the denominator in (5.2.11) increases, the numerator also increases resulting in a non-linear behavior. However, the positive effect predominates in most cases. This is exploited by Alg. 3 to find a value of $t_{min,i}$ that maximizes reliability p_i — starting from a minimum required value — for a given set of n nodes.

Alg. 3 is based on the following principles: First, a node j with a short deadline d_j and consequently a short $t_{min,j}$ produces a high m_{ij} for node i with a long d_i. This is because the short $t_{min,j}$ fits multiple times in $t_{max,i} - t_{min,i}$. Conversely, only one packet of node i is fit in $t_{max,j} - t_{min,j}$, i.e., $m_{ji} = 1$ implying $t_{min,i} > t_{max,j} - t_{min,j}$. Second, as will be later shown in Section 6.3.2 that a greater m generally leads to less reliability. As a result, Alg. 3 limits any node j's own m_{jj} to be equal to 1, which means that $t_{min,j} \geq \frac{t_{max,j}}{2}$ has to hold as per (5.2.8). All other m_{ij} for $i \neq j$ will be greater than 1 for any node i with $d_i > d_j$ and equal to 1 for $d_i < d_j$.

Alg. 3 starts by sorting nodes in non-decreasing order of d_i. It then calculates $t_{max,1}$ and $t_{min,1}$ of the node with the shortest deadline, where $t_{min,1}$ is set to $\frac{t_{max,1}}{2}$ with $m_{11} = 1$ as mentioned above. By setting $t_{min,1}$ to the lowest possible value, the interval $t_{max,1} - t_{min,1}$ are maximized as well as p_1 — see (5.2.11). This $t_{min,1}$ is checked for validity using (5.2.12) in line 4 assuming $m_{1j} = 1\,\forall j$, since $d_1 \leq d_j$ holds for $2 \leq j \leq n$. Clearly, in the case that (5.2.12) does not hold for the chosen $t_{min,1}$, the algorithm will exit with an error.

From line 7 onwards, Alg. 3 iterates over the remaining nodes computing $t_{max,i}$ in line 8 and $t_{min,i}$ in line 9 to 21. More specifically, $t_{min,i}$ is calculated by gradually incrementing m_{i1} and subtracting $m_{i1} \cdot t_{min,1}$ from $t_{max,i}$. This reduces $t_{min,i}$ in steps of $t_{min,1}$ making (5.2.11)'s denominator greater. A greater m_{i1} also increases (5.2.11)'s numerator, however, the positive effect predominates and p_i improves, i.e., it increases, in most cases.

The value of $t_{min,i}$ obtained this way has to be checked for validity. To this end, Alg. 3 first calculates all m_{ij} in line 13 by either using (5.2.8) for $j < i$, i.e., for the nodes with shorter deadlines, or setting $m_{ij} = 1$ for all $j > i$ according to the above discussion. In line 14, the value of $t_{min,i}$ is checked to be between the bounds given by (5.2.12) and $\frac{t_{max,i}}{2}$ respectively. If this is not the case, the program exits with an error when $m_{i1} = 1$, i.e., there was no previous value of m_{i1} for which a valid $t_{min,i}$ could be found. Otherwise, it continues with the next node i until valid $t_{min,i}$ have been found for all i or the program exists with an error.

Algorithm 3 Optimizing $t_{min,i}$ for each node i

```
Require: set of nodes with $l_i$, $d_i$, $k_i$, minimum required $p_i$
Require: $n$
 1: sort nodes in order of non-decreasing deadlines $d_i$
 2: $t_{max,1} = \frac{d_1 - l_1}{k_1}$
 3: $t_{min,1} = \frac{t_{max,1}}{2}$
 4: if (5.2.12) does not hold for $i = 1$ and $m_{1j} = 1 \forall j$ then
 5:    return (not feasible)
 6: end if
 7: for $i = 2$ to $n$ do
 8:    $t_{max,i} = \frac{d_i - l_i}{k_i}$
 9:    $m_{i1} = 0$
10:    while 1 do
11:       $m_{i1} = m_{i1} + 1$
12:       $t_{min,i} = t_{max,i} - m_{i1} \cdot t_{min,1}$
13:       $m_{ij} = \left\lceil \frac{t_{max,i} - t_{min,i}}{t_{min,j}} \right\rceil \forall j < i$ and $m_{ij} = 1 \forall j > i$
14:       if (5.2.12) does not hold or $t_{min,i} < \frac{t_{max,i}}{2}$ then
15:          if $m_{i1} = 1$ then
16:             return (not feasible)
17:          else
18:             restore last (valid) values of $m_{ij}$ and $t_{min,i}$
19:             break
20:          end if
21:       end if
22:    end while
23: end for
24: return (feasible)
```

In Section 6.3.2, a detailed numerical analysis is performed to visualize and further examine the impact of arbitrary deadlines and packet lengths on reliability. Next, practical factors are included, whereby Section 5.3 starts with external interference.

5.3. External interference

While the previous sections extended the presented model to allow for arbitrary node types with different deadlines and packet lengths, this section analyzes the effects of external interference on the RARE protocol, i.e., how noise from outside the network affects the system's performance. For a general overview of external interference see Section 2.2.2.

The following analysis is based on the models and assumptions from Section 1.3, in which external interference is mathematically described by a *duty cycle* σ, i.e., the largest possible ratio of pulse width to inter-pulse separation of the noise. Note that for simplicity, it is assumed that σ is the same for all nodes in the network, i.e., these are all affected equally by interference.

This σ gives the greatest probability of encountering an external interference pulse at the communication channel [87] — note that $\sigma \leq 1$ holds. Clearly, this probability is independent of q in (5.1.3), i.e., the probability of packet loss due to internal interference, since external interference is independent of any of the (internal) nodes in the network. As a result, when considering external interference, the probability of packet loss is given by:

$$\hat{q}_i = q_i + \sigma - q_i \cdot \sigma = q_i + (1 - q_i) \cdot \sigma, \tag{5.3.1}$$

i.e., the probability that a packet is lost at the channel either by internal or by external interference. Note that the binomial distribution can still be applied, since $\hat{q}_i$ in (5.3.1) is independent of the node and the packet being sent. As a result, proceeding as for the case with no external interference, the probability that k_i consecutive packets are lost is:

$$1 - p_i = (q_i + \sigma - q_i \cdot \sigma)^{k_i}\,, \tag{5.3.2}$$

and, hence, replacing q_i as per (5.2.10) this can be solved for $t_{min,i}$ such that p_i, the desired reliability requirement, is satisfied under external interference:

$$t_{min,i} \leq t_{max,i} - \frac{\left(l_i\left(\sum\limits_{j=1;j\neq i}^{n} m_{ij}\right) + \sum\limits_{j=1;j\neq i}^{n} m_{ij}\cdot l_j\right)\cdot(1-\sigma)}{\sqrt[k]{1-p_i} - \sigma}. \tag{5.3.3}$$

If $t_{min,i}$ becomes negative or greater than $t_{max,i}$, it will not be possible to fulfill the reliability requirement p_i for the given external interference σ. The other constraints on $t_{min,i}$, i.e., (5.2.8) — or (5.4.1) when considering clock drift — still have to be satisfied. The value of $t_{max,i}$ is again given by (5.2.6) — or (5.4.2) when accounting for clock drift.

In summary, (5.2.6) and (5.2.12) can be used in the absence and (5.4.2) and (5.3.3) in the presence of external interference. In the next section, the presented theory is further extended to account for clock drift.

5.4. Clock drift

All clocks used in electronic devices show a deviation in frequency with respect to each other, i.e., they *count* time at different rates. This deviation is known as clock drift and normally depends on a number of different factors such fabrication-induced variability, operating temperature, etc. — more on this in Section 2.2.2. As a result, since nodes do not synchronize, they will unavoidably have different time scales. To account for such effects, this section includes clock drift into the model of RARE.

Recall that each node i generates random inter-packet times t_{ix} in the interval $[t_{min,i}, t_{max,i}]$ — with a uniform distribution — and waits for these t_{ix} before sending any packet. A clock drift does not affect the generation of random inter-packet times, since bounds $t_{min,i}$ and $t_{max,i}$ are computed off-line. In other words, the length of the interval $[t_{min,i}, t_{max,i}]$ remains constant and, hence, (5.2.10) is still valid. However, since a node *counts* for t_{ix} before sending a packet, a clock drift leads to an *absolute* waiting time $\bar{t}_{ix}$ different than t_{ix}, i.e., the time without clock drift. As a result, this needs to be considered when selecting the bounds t_{min} and t_{max} for the random inter-packet times.

Towards this, recall that a node j may send m_{ij} packets in an interval of length $t_{max,i} - t_{min,i}$ of another node i. Again, node j waits for a random t_{jx} before sending each packet where t_{jx} is always greater than or equal to $t_{min,j}$. In an extreme case, it may happen that node j waits for $t_{min,j}$ time before each of the above mentioned m_{ij} packets. In this case, clock drift needs to be considered to guarantee that at most m_{ij} packets be in an interval of length $t_{max,i} - t_{min,i}$, otherwise (5.2.10) will stop being valid. To this end, let us denote by $\Delta t_{min,j}$ the maximum possible clock deviation (with respect to an ideal, non-drifting clock) in an interval of length $t_{min,j}$. As a result, clock drift can be incorporated into (5.2.8):

$$\begin{aligned}
m_{ij} &= \left\lceil \frac{t_{max,i} - t_{min,i}}{t_{min,j} - \Delta t_{min,j}} \right\rceil, \\
m_{ij}\cdot(t_{min,j} - \Delta t_{min,j}) &\geq t_{max,i} - t_{min,i}, \\
t_{min,j} &\geq \frac{t_{max,i} - t_{min,i} + m_{ij}\cdot \Delta t_{min,j}}{m_{ij}}.
\end{aligned} \tag{5.4.1}$$

Similarly, $t_{max,i}$ is selected such that it can guaranteed that k,i packets be sent within the specified deadline $d_{max,i}$. To consider clock drift in this case, let us denote by $\Delta t_{max,i}$ the maximum possible clock deviation in an interval of length $t_{max,i}$. In the worst case, a node may need to wait for $t_{max,i}$ before sending each of the k,i packets. Proceeding as follows to incorporate clock drift in (5.2.6) results in:

$$\begin{aligned} k_i \cdot (t_{max,i} + \Delta t_{max,i}) &\leq d_i - l_i \\ t_{max,i} &\leq \frac{d_i - l_i - k_i \cdot \Delta t_{max,i}}{k_i}. \end{aligned} \tag{5.4.2}$$

Note that this new bound for $t_{max,i}$ can be used instead of (5.2.6) to compute the $t_{min,i}$ that satisfies the reliability requirement p_i in (5.2.12). The value of $t_{max,i}$ in (5.4.2) needs to be considered in computing the other bounds on $t_{min,i}$ such as (5.2.8) and (5.4.1) taking clock drift into account.

Clearly, (5.4.1) and (5.4.2) requires knowing $\Delta t_{min,j}$ and $\Delta t_{max,i}$, which depend on $t_{min,j}$, $t_{min,i}$ and $t_{max,i}$. However, clock deviation due to clock drift will increase with the length of the considered time interval. Since $t_{min,i} < t_{max,i} < \hat{t}_{max,i}$ holds for $\hat{t}_{max,i} = \frac{d_i - l_i}{k_i}$, it is that $\Delta t_{min,i} < \Delta t_{max,i} < \Delta \hat{t}_{max,i}$ also holds where $\Delta \hat{t}_{max,i}$ is the maximum clock deviation in an interval of length $\hat{t}_{max,i}$. As a result, to resolve the above dependency, $\Delta t_{min,j}$ and $\Delta t_{max,i}$ can be safely replaced by $\Delta \hat{t}_{max,j}$ and $\Delta \hat{t}_{max,i}$ respectively in (5.4.1) and (5.4.2).

5.5. Key findings

This chapter presented RARE, a probabilistic MAC protocol designed to offer high reliability in low-cost, unidirectional networks. Unlike DEEP, which guarantees full reliability, i.e., data always reaches its destination in the worst case, RARE makes use of the fact that many applications can tolerate some data loss. In other words, it can be configured to a user-specified reliability of less than 100 %, i.e., to a certain probability that data reaches its destination within a deadline in the worst case. Towards this, nodes send their data as sequences of k redundant packets with random inter-packet times selected from an interval $[t_{min}, t_{max}]$. By adjusting t_{min}, t_{max} and k, reliability can be adapted to the given requirements.

Tolerating packet losses significantly reduces pessimism compared to DEEP's deterministic model and allows the use of a probabilistic model. This has the advantage that is reduces the model's complexity — probabilistic methods typically facilitate the modeling of complex dependencies and random effects that are difficult to express otherwise — and avoids the high overhead of full reliability, i.e., the large packet numbers and long inter-packet times. For example, in a network of $n = 30$ nodes, DEEP requires each node to send $k = 30$ packet per sequence for full reliability. RARE, on the other hand, can achieve a reliability $p = 99,98\%$ with only $k = 3$ packets when assuming the parameters from Section 6.1. This is a 10-times reduction in energy consumption — energy is directly proportional to the number of packets sent — while the chance of losing data is only $0,02\,\%$ in the worst case. More on this later in Chapter 6.

Compared to other protocols from the literature, RARE is designed for heterogeneous networks, i.e., it supports arbitrary node types with different deadlines and packet lengths. This has the advantage that packet loss probabilities can be modeled more accurately, which in return improves reliability and the maximum achievable network sizes. In addition, the presented theory further includes the effects of practical factors such as external interference and clock drift, and presents an algorithm (of heuristic nature) to find an optimum configuration of MAC parameters, such that reliability is maximized for a given WSN.

In summary, RARE presents a MAC protocol for unidirectional WSNs that enables high reliability at very low complexity and energy consumption. Note that, in Appendix B.3, RARE is further extended to bidirectional communication. This extension, called bi-RARE, implements carrier sensing and ACKs to further enhance the average performance of the network. Next, the following section evaluates DEEP and RARE in a numerical analysis and simulation-based experiments.

Chapter 6.

Evaluation and results

This chapter evaluates the DEEP and RARE protocols and compares them with other state-of-the-art approaches from the literature. To this end, this chapter consists of two parts: a numerical analysis and a simulation-based evaluation. The numerical analysis evaluates the worst-case behavior of the protocols and examines the influence of certain MAC parameters on the performance of the network. The simulation-based evaluation, on the other hand, examines the average performance of the protocols and further compares them to other MAC solutions from the literature.

In order to obtain suitable values for the following experiments, an exemplary case study is introduced next. This describes a typical WSN application from Industry 4.0 and further illustrates the different steps of network design. Note that this application has been selected, because both DEEP and RARE can be implemented for it. However, any other WSN could have been used as well, for example, a home automation setting as described in [68]. For a more detailed list of possible application areas, see Section 7.1.

6.1. Use Case – Intelligent assembly line

Modern assembly lines are expected to increasingly rely on mobile robots, as these can perform tasks quickly and efficiently. These robots are usually limited to a set of pre-programmed tasks and lack autonomy. As a result, they still require human workers for manipulating fragile parts or taking common sense decisions leading to a shared physical space.

This, of course, requires a high degree of safety, since weighty robots can potentially harm humans and/or other robots. For this reason, humans and robots are equipped with sensors to locate and track their positions. This data is periodically broadcast and received by a central control station that computes mobile robots' trajectories. If robots comes close to humans or other robots, i.e., at a *safety distance*, these are slowed down or even stopped to avoid accidents. In the case of communication loss, fail-save mechanisms need to be specified to guarantee safety. For example, robots may perform an emergency stop, if no updates from the central control station are received within a specified period of time, etc.

6.1.1. Timings and data structures

Workers and mobile robots, i.e., transmitting nodes in the network, periodically broadcast their current position and speed with a total of 10 bytes. For this they are equipped with a wireless transceiver that handles transmissions and receptions of data packets, which have a basic frame format of 8 bytes preamble, 2 bytes Start Frame Delimiter (SFD) and 2 bytes Cyclic Redundancy Check (CRC). Assuming a transmission speed of 2 Mbit/s, the resulting length of a single data packet is:

$$\begin{aligned} l_{max} &= T_{preamble} + T_{SFD} + T_{payload} + T_{CRC} \\ &= 32\,\mu s + 8\,\mu s + 40\,\mu s + 8\,\mu s = 88\,\mu s. \end{aligned}$$

In addition, there will be a delay at the central control station, which is receiving all data packets. This delay, denoted by t_{host}, accounts for computation of trajectories, collision avoidance algorithms, and the reaction time of actuators and is estimated to be in the order of several tens of milliseconds. In the following, it is assumed that $t_{host} \leq 80\,ms$.

6.1.2. Physical world

This example considers moving robots with a maximum speed of $10\,km/h$, i.e., around $3\,m/s$, and further assumes that the walking speed of human workers is no more than $5\,km/h$, i.e., around $1.5\,m/s$. In addition, robots have a stopping distance of $0.5\,m$ (from maximum speed to zero) and their safety distance is fixed to a radius of $4\,m$.

In the worst-case, two robots move at full speed towards each other, which results in a relative speed of $6\,m/s$. Since it has to be guaranteed that both robots stop on time to avoid collisions, the central control station has to receive their packets within:

$$t_{WC} = \frac{4\,m - 0.5\,m}{6\,m/s} = 580\,ms,$$

from the time they enter each others safety distance. Subtracting t_{host} as a safety tolerance and to account for processing times, an upper bound on communication delay can be obtained:

$$d_{max} = t_{WC} - t_{host} = 500\,ms.$$

This is the time by which data must be received at the central control station to avoid triggering fail-save mechanisms and cause an emergency stop.

6.1.3. Remaining network parameters

The previous sections determined a packet length $l_{max} = 88\,\mu\text{s}$ and a deadline $d_{max} = 500\,\text{ms}$, however, there are other parameters to be defined for the experiments in the following sections. To this end, it is assumed that each node is equipped with a CC2545 [83] transceiver IC for handling data transmissions and receptions. The transceiver switching time, i.e., this is the time required by a node to switch between receive and transmit mode, is consequently $t_{switch} = 130\,\mu s$ and the sensitivity, i.e., the time required to detect the channel as busy, is set to $\bar{t} = 10\,\mu\text{s}$. Carrier sensing has been fixed to $t_{sen} = 150\,\mu s$, i.e., slightly longer by than t_{switch} to be able to detect the *gap* between data packet and ACK — see Appendix B.1 for more details. Lastly, ACKs have the same frame format as data packets, but a reduced payload of 2 bytes containing the source node address. Their length is consequently $l_{ACK} = 56\,\mu s$.

Note that the above parameters are common to all protocols in the following experiments, however, there are further MAC-specific values that need to be defined. Towards this, the next section provides an overview of all protocols under comparison and describes their working principles and remaining parameters.

6.2. Protocols under comparison

This section provides an overview of the different MAC protocols that are compared in the following experiments. These are:

- **DEEP** is the protocol proposed in Chapter 4 that is designed to enable fully reliable communication in unidirectional networks. Unless otherwise noted, the heuristic model from Section 4.2 is used.
- **RARE** is the protocol proposed in Chapter 5 that is designed to guarantee a configurable level of reliability in unidirectional networks.
- **bi-DEEP** is an extension of DEEP for bidirectional networks that uses carrier sensing and ACKs — see Appendix B.2.

- **bi-RARE** is an extension of RARE for bidirectional networks that uses carrier sensing and ACKs — see Appendix B.3.
- **eMAC** (extensive MAC) from [10] [11] is a deterministic protocol similar to DEEP that enables fully reliable communication in unidirectional networks.
- **TDMA** (time division multiple access) is a synchronous protocol, in which nodes transmit during dedicated time slots to avoid collisions. For this, they share a common clock by periodically receiving synchronization beacons from the sink.
- **CSMA** (carrier sense multiple access) is an asynchronous protocol that implements non-persistent carrier sensing and a random backoff scheme, similar to IEEE 802.15.4.

The eMAC protocol [10] [11] is based on the same principle as DEEP and bi-DEEP, i.e., it consists in nodes sending their data as sequences of redundant packets with carefully chosen inter-packet times. This way, unique transmission patterns are generated, ensuring that at least one packet of a sequence reaches its destination in the worst case. The main difference to DEEP is the methodology for selecting inter-packet times, which yields different results. While DEEP-analytic (Section 4.1) uses formulas to calculate inter-packet times and DEEP-heuristic (Section 4.2) implements a heuristic search algorithm, eMAC uses Integer Linear Programming (ILP) for calculations. This results in very short and multiple values of inter-packet times per node, which reduces delays and enables larger network numbers. However, due to the high complexity of the problem, only small network with $n \leq 6$ nodes could be calculated by the authors [10]. To solve this problem and allow for larger n, the same authors introduced a heuristic algorithm in [11], which greatly reduces computation times, however, at the cost of lower performance.

Note that eMAC has been selected for comparison, since it follows a similar strategy as DEEP and is applicable for purely unidirectional networks without the need for special transceiver hardware. In contrast, other unidirectional MACs such as [14], [51], and [95] (see Chapter 3) fail to meet these criteria and, hence, have not been included in the following experiments.

Similarly, CSMA and TDMA have been selected for comparison, as these are commonly-used for WSNs (see Section 2.5). However, since there exists a many different modified versions in the literature, as discussed in Chapter 3, the following experiments only consider baseline CSMA and TDMA; these are also the core technologies of most alternative approaches [33] [78] [99]. In particular, during high contention — which the following experiments simulate — most of these alternative approaches fall back to the performance of either CSMA or TDMA and are, hence, not further considered in the presented comparison. For example, [33] [78] reduce to CSMA and [99] reduces to TDMA at high congestion — see also Chapter 3.

In the TDMA scheme, communication is organized in cycles. Each such cycle starts with a beacon message for synchronization, followed by n dedicated time slots for data transmission, where n is the number of nodes to transmit in the current cycle — see Section 2.5.1. In the following experiments, the beacon is a normal data packet, which contains synchronization and network information, having a length of $88\,\mu s$. Time slots have a length of $404\,\mu s$, i.e., they contain a data packet, an ACK and processing and switching times. In addition, a guard time interval t_{guard} is added before every slot and beacon to account for clock drift.

For increased robustness, there are three TDMA cycles within a deadline, which allows nodes to retransmit packets or receive another beacon, if the previous ones were corrupted. These cycles are evenly spaced within the deadline, i.e., these start at times 0, 167 ms and 333 ms from the beginning of d_{max}. Lastly, the beacon rate needs to be configured, i.e., the time after which a node has to receive a beacon to resynchronize before its clock drift becomes too large and it starts violating slot boundaries. Assuming a standard oscillator with $100\,ppm$, the beacon rate will mainly depend on t_{guard}. The greater the beacon rate, the shorter t_{guard} can be, resulting in short delays. However, since delay is generally less important as long packets arrive before d_{max}, we select t_{guard} to be as long as possible to lower the beacon rate and decrease energy consumption. That is, it is meaningful to select a beacon rate of 3.5 s and $t_{guard} = 700\,\mu$s. Note that the sink sends a beacon in every cycle, but sensor nodes only listen for them if they need to synchronize, i.e., every 3.5 s.

The CSMA scheme is based on the non-persistent, non-slotted CSMA protocol as defined in IEEE 802.15.4 — see Section 2.5.2. Nodes use carrier sensing to assess the channel state prior to packet transmissions and only send if the channel is free. In case the channel is blocked or whenever a transmission failed, i.e., no ACK was received, the node performs a random backoff before trying to retransmit. The backoff time is calculated using the binary exponential backoff formula $\text{rand}(1, 2^c - 1) \cdot t_{base}$ with c being the number of failed transmissions. According to IEEE 802.15.4, the maximum backoff and retransmission numbers are set to 4 and 3 respectively and a base multiplier of $t_{base} = 320\,\mu$s is used. The contention window — a value which defines the minimum and maximum length of the backoff time — was increased to [32, 1024] to prolong backoff times and allow CSMA to use the full deadline (with default parameters as per IEEE 802.15.4, only a short fraction of the deadline is used). This increases the overall performance of CSMA in the simulation and, in particular, reduces collision rates and energy consumption.

6.3. Numerical analysis

This sections presents a numerical analysis that evaluates the worst-case performance of the proposed protocols DEEP and RARE and examines the influence of certain MAC parameters on the performance of the network, i.e., how changing them affects the protocol's behavior.

The following experiments are based on the parameters obtained from the intelligent assembly line in Section 6.1. Further, the number of nodes, i.e., workers and mobile robots, is set to $n = 30$ and in the case of RARE, reliability is set to $p = 99.999\%$ (equal to a sequence loss rate $1 - p$ of 10^{-5}). For simplicity, external interference and clock drift are neglected for now, as this is discussed later in Section 6.4.4 and 6.4.5.[1] However, observations and conclusions remain valid.

In particular, the following experiments evaluate the performance of the proposed MACs by analyzing the maximum possible network size n_{max}, i.e., the number of nodes that can be safely accommodated in a network for the given parameters. This n_{max} results from delay, channel utilization and throughput of the MAC and often constitutes the limiting factor for many applications. The higher this n_{max}, the better the corresponding MAC is.

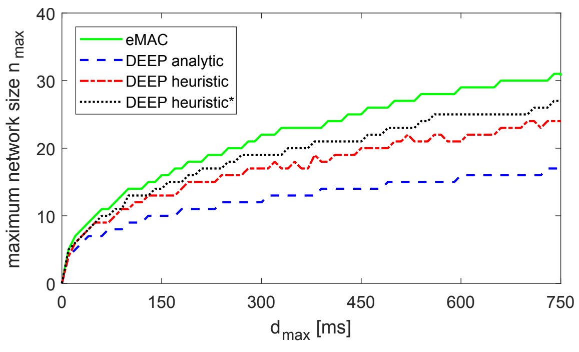
6.3.1. The DEEP protocol

This section focuses on analyzing the worst-case behavior of DEEP. In particular, the different network models are examined more closely, i.e., DEEP-analytic from Section 4.1 and DEEP-heuristic from Section 4.2, and their differences in performance, computing time, etc. are evaluated. Lastly, this section also discusses the impact of arbitrary node types with different deadlines and packet lengths on performance.

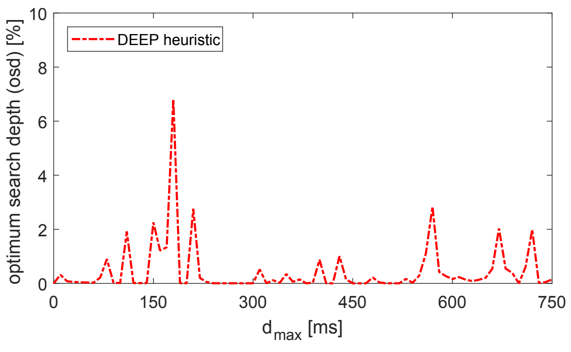
Maximum number of nodes versus deadline

First, let us analyze n_{max} for the case of both the deadline and packet size being the same for all nodes, as displayed in Fig. 6.1a. Clearly, for an increasing deadline, the maximum number of nodes n_{max} increases with all MACs, since longer deadlines allow sending more

[1]This assumption facilitates the understanding of results, since these are not distorted by additional errors.



(a) n_{max} versus deadline



(b) The *optimal search depth* (*osd*) of DEEP-heuristic that leads to DEEP-heuristic*.

Figure 6.1.: Maximum network size n_{max} vs. deadline and the corresponding *optimal search depth* (*osd*).

packets with less collisions. The greatest number of nodes can be realized using the eMAC scheme, since it has the most elaborate search optimization and allows multiple inter-packet times per single node. It is, however, computationally very expensive — as discussed later — and yields only around 15 % better results than DEEP-heuristic. For a deadline of 750 ms, it allows for around 31 nodes, whereas DEEP-heuristic can reach approximately 27 nodes. The DEEP-analytic approach only allows 17 nodes for this deadline.

The heuristic model of DEEP is depicted twice in Fig. 6.1a: The DEEP-heuristic curve depicts the results from the unmodified algorithm as shown in Alg. 1, whereas the DEEP-heuristic* curve shows a slightly optimized variant. To explain the difference between them, let us again consider Alg. 1. This algorithm gradually searches for periods for each node and takes the first value that is proven valid by function *check_period()*. However, this first choice can be further optimized. In particular, it can be observed in Fig. 6.1a that DEEP-heuristic sometimes finds a greater n_{max} for a smaller deadline. This suggests making Alg. 1 search for a local optimum in the close vicinity of a deadline, which leads to the DEEP-heuristic* curve in Fig. 6.1a. That is, for a given deadline d_i, the local optimum n_{max} is searched in $[d_i \cdot (1 - osd), d_i]$, where $osd \in [0, 1]$ denotes what is called the *optimum search depth*. This gives the length of the search interval in relation to d_i: an $osd = 0$ means no search for an optimum; an $osd = 1$ means a full search in $[0, d_i]$. Clearly, the greater the value of osd, the more computation time is required.

Fig. 6.1b shows the values of osd that leads to a locally optimum n_{max}. As it can be noted, an $osd = 0.07$ is the greatest osd required to find the optimum n_{max} at a 180 ms deadline. On the other hand, for other deadlines, the osd peaks to around 0.02 to 0.03, i.e., 2 % to 3 % of the deadline, as illustrated in Fig. 6.1b.

Computation time

Let us now discuss the computation overhead by the different approaches, i.e., the time that is needed to calculate inter-packet times for all nodes. All calculations were performed on an Intel i7 3520M processor at 3.9 GHz and with 8 GB of memory. Results are depicted in Fig. 6.2.

As expected, the DEEP-analytic scheme has the lowest computation time, since inter-packet times can be calculated analytically with linear complexity. Its computation time for 30 nodes is below 5 ms. DEEP-heuristic, i.e., without optimization, offers the second lowest computation time. Its computation time slowly rises and is around 330 ms for 30 nodes. With optimization and an osd of 10 % for the DEEP-heuristic* variant, the computation time rises to 12 s for 30 nodes.

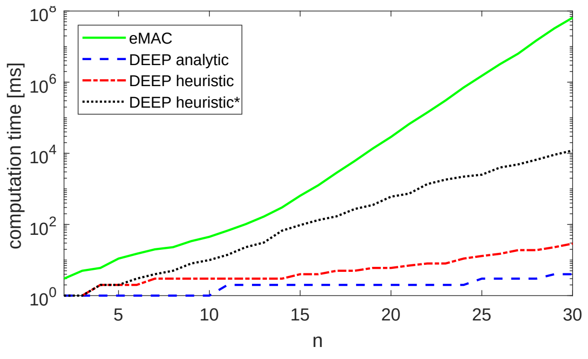


Figure 6.2.: Computation time versus number of nodes

In case of eMAC, the multiple inter-packet times per single node and the extensive optimization lead to considerably higher computation overhead. For 30 nodes, its computation time is around 18 hours as shown in Fig. 6.2. Although these algorithms normally run off-line on a fast computer, high computation times negatively impact testing and debugging of a system, in particular, during the production and early deployment phase.

Note that, although not depicted in Fig. 6.2, the RARE protocol features a similarly low computation time as DEEP-analytic. This is because its parameters can also be calculated via formulas without the need of a heuristic search algorithm. In the above example, the computation time for RARE was around 1 ms for $n = 30$.

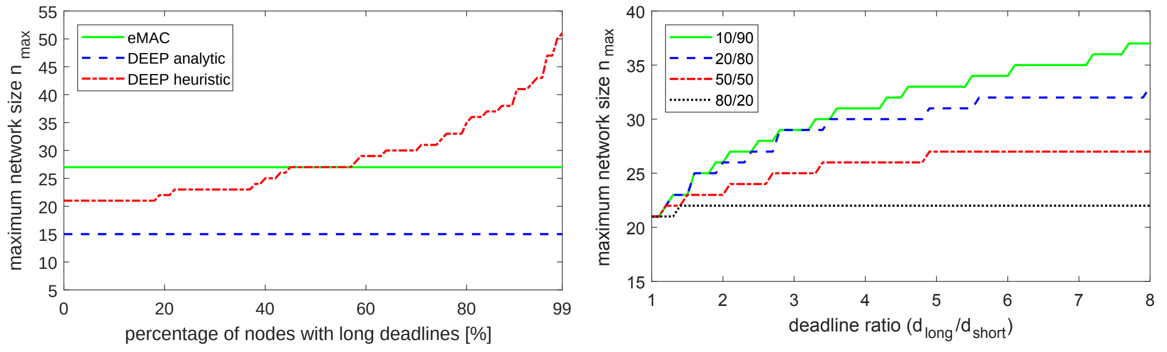
Effect of different deadlines

The following experiments examine how n_{max} is influenced when nodes with different deadlines are considered. Fig. 6.3a shows an exemplary system with two types of nodes: fast nodes with a deadline d_{short} of 500 ms and slow nodes with a deadline d_{long} of 10 s.

Since both eMAC and DEEP-analytic are not designed for systems with different deadlines, they cannot take advantage of the long deadline and have to assume the short deadline for all nodes. Their n_{max} consequently do not change for an increasing number of nodes with long deadlines and stay at the value of 0 % slow nodes (i.e., only fast nodes). In contrast, DEEP-heuristic allows a higher n_{max} for a rising percentage of nodes with long deadlines, since it was specially designed to benefit from this. For 50 % or more slow nodes, DEEP-heuristic results in the greatest n_{max} — see Fig. 6.3a, even greater than eMAC with its considerably more elaborate optimization.

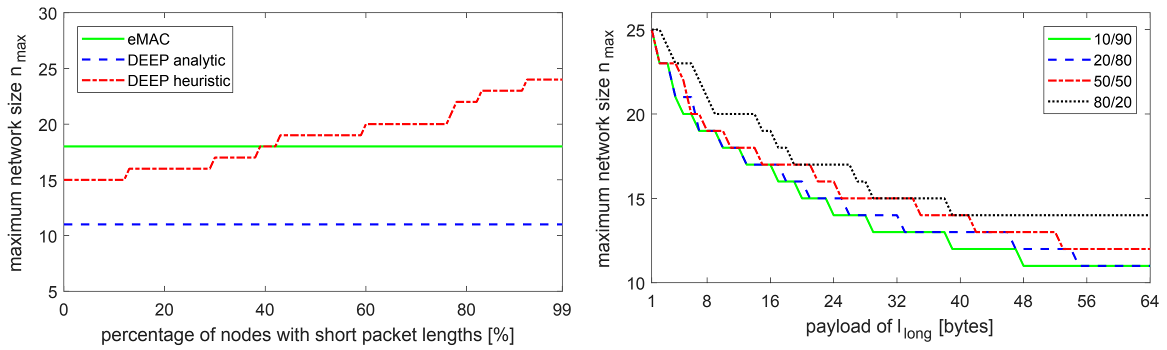
In Fig. 6.3a, the deadlines d_{short} and d_{long} are fixed to $d_{short} = 500$ ms and $d_{long} = 10$ s, i.e., a deadline ratio d_{long}/d_{short} of 20. Since this ratio also influences the maximum number of nodes n_{max} by DEEP-heuristic, this is further investigated in Fig. 6.3b. Here, the system is again composed of two node types: fast nodes with a deadline d_{short} of 500 ms and slow nodes with a deadline d_{long}, which in this case, can be varied from 500 ms (ratio of 1) to 4 s (ratio of 8). The number of fast nodes compared to the number of slow nodes within the system is fixed, e.g., the 20/80 system is composed of 20 % fast nodes and 80 % slow nodes. As depicted in Fig. 6.3b, the greater the amount of nodes with long deadlines in the system, the higher n_{max} is for an increasing d_{long}/d_{short} ratio. However, n_{max} saturates for high deadline ratios, i.e., making d_{long} greater has no positive effect on n_{max} from a certain point onwards.

This saturation point is reached earlier, i.e., for smaller deadline ratios, the lower the number of slow nodes is. That is, systems with a small amount of slow nodes, for example, the 80/20 system with 20 % slow nodes, have little benefit from a greater d_{long}. Systems with a high number of slow nodes, for example, the 10/90 system with 90 % slow nodes, benefit from greater values of d_{long}.



(a) Varying ratios of fast nodes with a short deadline $d_{short} = 500\,\text{ms}$ and slow nodes with a long deadline $d_{long} = 10\,\text{s}$.

(b) Varying amounts of fast/slow nodes for different d_{long} that change from 500 ms (ratio of 1) to 4 s (ratio of 8)



(c) Varying ratio of slight nodes with a short packet length $l_{short} = 52\,\mu\text{s}$ and intense nodes with a four times longer packet length $l_{long} = 208\,\mu\text{s}$.

(d) Varying amounts of slight/intense nodes and different payloads of l_{long}.

Figure 6.3.: n_{max} for a system consisting of two node types: nodes with short and long deadlines (i.e., fast/slow nodes) or nodes with short and long packet lengths (i.e., slight/intense nodes). Note that the ratio between these are given in percent, e.g., 20/80 means 20 % fast/slight nodes and 80 % slow/intense nodes.

The reason is that, in the worst case, a fast node with a short deadline can be activated multiple times within the longer deadline of a slow node. This means that the slow node can potentially collide with more packets of the fast node within its long deadline. This effect increases with the number of fast nodes and the ratio of d_{long}/d_{short} — as shown in Fig. 6.3b — leading to the observed saturation.

Effect of different packet lengths

Let us now analyze the effect of different packet lengths on n_{max}. To this end, Fig. 6.3c regards a system composed of two node types: slight nodes with short packet lengths $l_{short} = 52\,\mu\text{s}$ corresponding to one byte payload and intense nodes with a four times longer packet length $l_{long} = 208\,\mu\text{s}$ corresponding to a 40-byte payload.[2]

Varying the percentage of nodes with short deadlines does not influence n_{max} of DEEP-analytic and eMAC, as both of them are not designed for different packet lengths and, therefore, have to assume l_{long} for all nodes. In contrast, as illustrated in Fig. 6.3c, DEEP-heuristic leads to improved results when combining different packet lengths. The values of n_{max} increase as the percentage of nodes with a short packet lengths increases, outperforming the

[2]As discussed before in Section 6.1.1, every transmission contains a preamble, SFD, etc., which adds 48 μs additional time to each packet length.

eMAC protocol from 40 % slight nodes — with short packet lengths — onwards. Note that long packets leads to a higher channel utilization and, hence, n_{max} decreases for all algorithms. That is, values of n_{max} in Fig. 6.3c are smaller than those of Fig. 6.3a although curves look similar.

Further, n_{max} is also influenced by the packet length ratio between different nodes, i.e., how big the packet length of a node is compared to another node. In Fig. 6.3d, similar to Fig. 6.3c, a system with two different node types is used: slight nodes with a short packet length l_{short} corresponding to a payload of one byte and intense nodes with a long packet length l_{long} that can be varied between 1 byte (ratio of 1) and 64 bytes payload (ratio of around 6). Now, the amount slight compared to intense nodes is fixed, e.g., the 20/80 curve is a system with 20 % slight and 80 % intense nodes.

The negative impact of the packet size on n_{max} is clearly visible: As the packet length l_{long} rises, the channel utilization increases. A high channel utilization implies less time for other nodes to send their packets, therefore, the maximum number of nodes n_{max} decreases in a considerable manner.

When l_{long} has a payload of 1 byte, the packet lengths are the same for both node types and all curves displayed in Fig. 6.3d have the same n_{max}. When the payload of l_{long} rises, the n_{max} values of the systems with more intense nodes decrease more rapidly, e.g., the 10/90 system with 90 % intense nodes falls deeper than the 80/20 system with only 20 % intense nodes. In contrast to Fig. 6.3b, note that there is no saturation effect for high packet length ratios, but n_{max} tends to 0. In summary, long packets should be avoided in order to achieve an acceptable n_{max}. Alternatively, the deadline of nodes with long packets should be increased to compensate this negative effect.

6.3.2. The RARE protocol

This section further analyzes the worst-case performance of RARE. Note that RARE is generally more complex to analyze compared to DEEP, since more parameters need to be configured. In the case of DEEP, packet numbers are fixed (to n) and inter-packet times solely depend on packet lengths and deadlines. However, in the case of RARE, these additionally depend on t_{min}, t_{max}, k and m, hence, a more detailed analysis is required.

Towards this, the following analysis evaluates the impact of RARE's parameters t_{min}, t_{max}, k, m and p by measuring the resulting performance in n_{max} and p, i.e., how many nodes are feasible for a given setting and what reliability can be achieved. Similar as before, it is first assumed that deadlines and packet lengths are the same for all nodes (denoted by d_{max} and l_{max}), which is then extended to arbitrary values later.

Selecting $\mathbf{t}_{max}$ and $\mathbf{t}_{min}$

Given the parameters n, l_{max}, and d_{max}, the first step is to determine the number of packets k that guarantees the desired reliability p. This implies finding valid values of t_{max} and t_{min} for a given k that not only satisfy (5.1.6) and (5.1.7), but also (5.1.1) and (5.1.4).

Fig. 6.4a and Fig. 6.4b show plots of (5.1.6) (solid line) and (5.1.7) (dashed line) for $p = 1 - 10^{-5}$ and different values of k. Upward- and downward-pointing triangles identify valid values of t_{max} and t_{min}. The value of m has been set to 1 in Fig. 6.4a and to 2 in Fig. 6.4b. Recall that this parameter determines the lower bound of t_{min} and, hence, the number of packets from the same node in an interval of length $t_{max} - t_{min}$.

From Fig. 6.4a and Fig. 6.4b, it can be observed that $p = 1 - 10^{-5}$ cannot be guaranteed for every k. In the case of $m = 1$ in Fig. 6.4a, the system is only feasible for k in $[6, 35]$. It should be noted that the number of valid values for t_{max} and t_{min} reaches its maximum for $k = 12$. For higher k this number decreases until having only one valid value for $k = 35$.

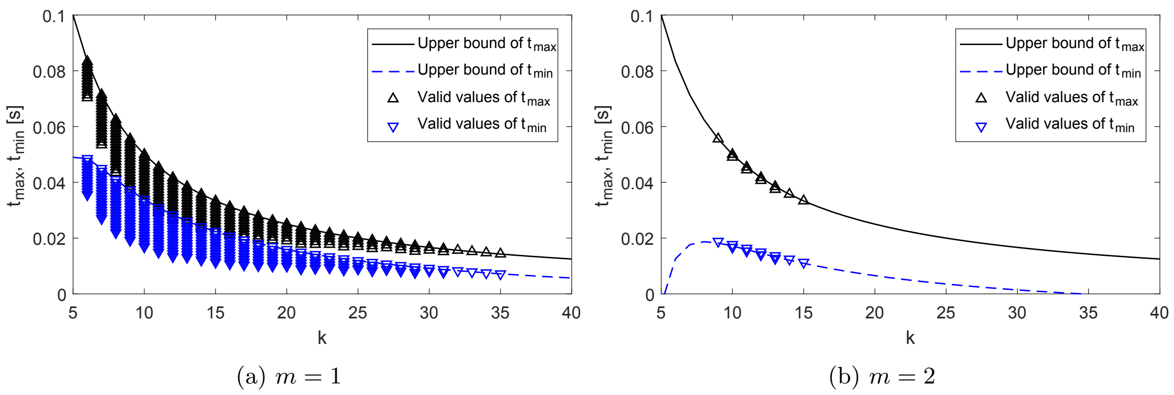


Figure 6.4.: Valid values of t_{max} and t_{min} for different m

That is, a higher k does not increase the reliability anymore, since the increasing interference between nodes starts to be the dominating effect from this point on.

Now, in the case of $m = 2$ in Fig. 6.4b, the system is only feasible for k in $[9, 15]$ and, in general, there are less valid values of t_{max} and t_{min}. This is analyzed next in detail.

Effect of m

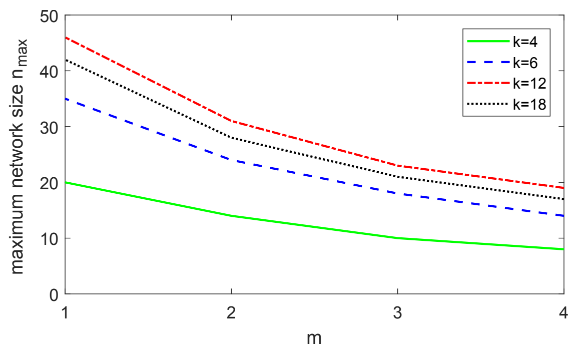
Let us now study the effect of m on the maximum number of nodes n_{max} that can be reached for a specified p. Fig. 6.5a shows how n_{max} varies with m and different values of k. For $k = 6$, $p = 1 - 10^{-5}$ and $m = 1$, around 35 nodes are possible, which reduces to 14 for $m = 4$. A similar behavior can be observed for other k.

In other words, m allows for more flexibility in selecting values of t_{min}. The greater m is chosen, the closer t_{min} may be to zero — see (5.1.1). As a result, the interval $[t_{min}, t_{max}]$ becomes longer decreasing the probability of packet collision — see (5.1.3). On the other hand, m also increases the number of packets in $[t_{min}, t_{max}]$ from the same node, which again increases the probability of packet collision — see again (5.1.3). In general, the second effect dominates such that a greater m negatively impacts the attainable size of the network. It is therefore meaningful to consider $m = 1$ and $t_{min} = \frac{t_{max}}{2}$ — minimizing the probability of packet collision for $m = 1$ — for the rest of this work.

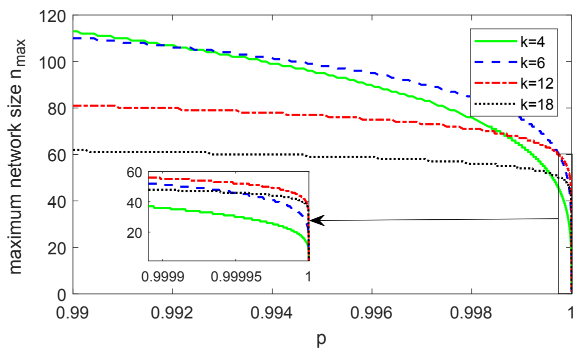
Reliability vs. number of nodes

Fig. 6.5b shows the dependency of the maximum possible number of nodes n_{max} on the desired reliability p for different values of k. It can be observed that sending a small number of packets, i.e., $k = 4$, allows an acceptably big n_{max} for low values of p. With $k = 4$ and $p = 0.95$, it is possible to achieve $n_{max} = 170$, but this rapidly reduces for very high p, for example to $n_{max} = 20$ for $p = 1 - 10^{-5}$. For these high reliabilities, more packets are required to achieve an acceptably high n_{max}. For example, for $p = 1 - 10^{-5}$, at least $k = 6$ are needed for achieving a network size of > 30 nodes. As mentioned previously, the only way of guaranteeing a reliability of 100 % ($p = 1$) with RARE is with only one node, i.e., $n = 1$.

For a given reliability value, there exists an optimal k that maximizes n_{max}. In Fig. 6.5b, for example, $k = 4$ performs best for p up to $p \leq 1 - 10^{-2}$ %. Similarly, $k = 6$ is optimal for $p \leq 1 - 10^{-4}$, $k = 12$ for $p \leq 1 - 10^{-6}$ and $k = 18$ for $p > 1 - 10^{-6}$. Note that if multiple values of k satisfy a given p and n, it is meaningful to select the lowest k, clearly, leading to less energy consumption.



(a) Different values of m and k



(b) Different values of p and k

Figure 6.5.: Maximum network size n_{max} for different m, p and k

Effect of arbitrary deadlines

Let us now analyze an exemplary system consisting of two node types in Fig. 6.6a: type 1 with short deadlines of $d_1 = 500\,ms$ and type 2 with varying deadlines of $500 \leq d_2 \leq 5000\,ms$. Again, for simplicity, the subindexes 1 and 2 represent the *node type* 1 and 2 respectively as defined above. The remaining parameters were set to $n = 30$, $k = 3$ and $l_1 = l_2 = 88\,\mu s$; m_{ij} are computed based on Alg. 3.

By modeling arbitrary deadlines, it is possible to reach higher reliabilities than in the case of assuming that all deadlines are the same. In this case, all deadlines are set to be equal to the shortest possible one in the system (see Section 5.1), which incurs pessimism as shown by the *ref* system in Fig. 6.6a. This allows for at most a reliability of 99.977 % for all d_2, which equals a sequence loss rate $1 - p \approx 2 \cdot 10^{-4}$. In case of the *20/80* system — 20 % of type 1 and 80 % of type 2 nodes — the reliability increases to around 99.9995 % and the sequence loss rate $(1 - p)$ decreases to $5 \cdot 10^{-6}$ for $d_2 = 5000\,ms$. This is an improvement of $(1 - p)$ by a factor of 40.

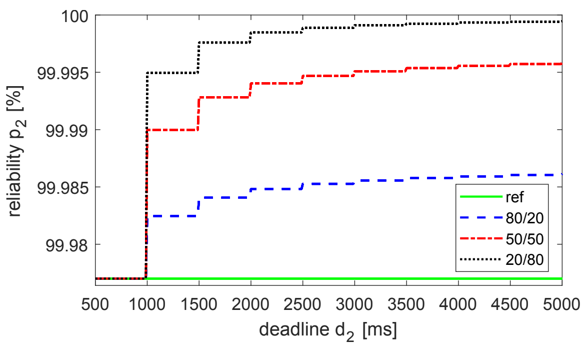
It can also be observed that the reliability p_2 increases stepwise every $500\,ms$ until slowly saturating for a large d_2. At every step, viz., when d_2 is a multiple of d_1, m_{i1} can be increased by one as per Alg. 3. This allows reducing the value of $t_{min,2}$ and, hence, results in a better reliability p_2 as shown in Fig. 6.6a. As expected, the more type 2 nodes there are in the system, the higher the value of p_2 that can be reached, since there will be more nodes with longer deadlines.

Effect of arbitrary packet sizes

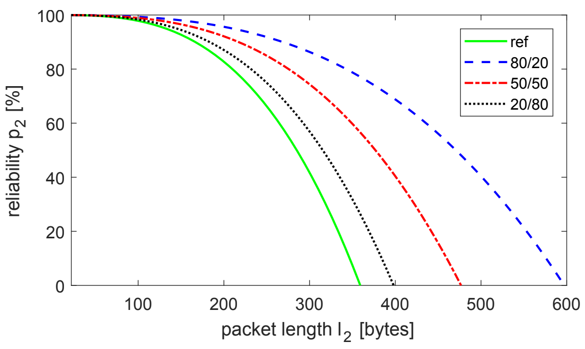
The following experiment analyzes the effect of arbitrary packet sizes on reliability. For this purpose, let us consider (5.2.4), which gives an upper bound on $t_{min,i}$. This bound depends on $t_{max,i}$ and $\Delta_{coll,i}$ as per (5.2.1). Now, by increasing the packet sizes l_i, the collision interval increases and, hence, $t_{min,i}$ decreases. However, $t_{min,i}$ is lower bounded by (5.2.2) and can therefore only decrease to a certain extent. Otherwise, the system stops being feasible.

Reliability is then conditioned by the packet sizes l_i of all nodes as per (5.2.1) and by the length of the shortest $t_{min,j}$ in the system — as per (5.2.2). In summary, allowing for arbitrary packet sizes instead of assuming the longest possible l_{max} for every node reduces $\Delta_{coll,i}$ and, hence, increases reliability according to (5.2.5).

Fig. 6.6b now displays an exemplary system with two node types: type 1 with short packets of $l_1 = 10$ bytes ($88\,\mu s$ as before in Sec. 6.1) and type 2 with varying packets of $10 \leq l_2 \leq 600$ bytes. For simplicity, the sub indexes 1 and 2 represent the *node type* 1 and 2 respectively and not directly the *node* 1 and 2. The remaining parameters were set to $n = 30$, $k = 3$, $d_1 = d_2 = 500\,ms$ and $m = 1$.



(a) For different deadlines: $d_1 = 500\,ms$, $500 \leq d_2 \leq 5000\,ms$; m_{ij} are computed by Alg. 3.



(b) For different packet lengths: $l_1 = 22$ bytes , $22 \leq l_2 \leq 600$ bytes.

Figure 6.6.: The maximum possible reliability p_2 for a system with two node types 1 and 2 and different deadlines (left) and packet lengths (right). The different curves show p_2 for different n_1/n_2-ratios, i.e., *20/80* means 20 % type 1 and 80 % type 2 nodes with $n_1 + n_2 = n = 30$. The *ref* curve shows the behavior of the basic scheme from Section 5.1, which does not allow modeling different deadlines/packet sizes.

As expected, an increasing l_2 leads to lower reliability, which reaches $p_2 = 0$ for high values of l_2, i.e., when (5.2.7) does not hold and, hence, the system is not feasible for any valid p_2. It can further be observed that a higher ratio of type 1 to type 2 nodes, i.e., when the number of type 2 nodes with their relatively big packets is low, allows for a higher reliability (of the remaining type 2 nodes in the system).

By taking arbitrary packet sizes into account, it is possible to reach higher reliabilities than for packets with the same size. In the latter case, the corresponding packet lengths are set to be equal to the longest possible packet in the system to guarantee the desired reliability (see Section 5.1), incurring pessimism as shown by the *ref* system in Fig. 6.6b. This system achieves a reliability of 65 % for $l_2 = 256$ bytes, whereas the *80/20* system — 80 % of type 1 and 20 % of type 2 nodes — allows for 91% reliability. This is an improvement of 40 % over the *ref* system from Section 5.1. Even for small differences in packet sizes, for example $l_2 = 44$ bytes, reliability can be improved considerably. In this case, p_2 can be increased from 99.81 to 99.9 %, which corresponds to a 50 % reduction of the sequence loss rate $(1 - p)$.

6.4. Simulation

Complementary to the previous section, where the worst-case behavior is illustrated in a numerical evaluation, this section further analyses the different MAC protocols with respect to their average performance. Towards this, a simulation was implemented in OMNeT++ [85] that runs different network parameters and records statistical values for very large numbers of transmissions — at least 1,000,000 packets were simulated for each of the presented curves.

The simulated network is based on the intelligent assembly line from Section 6.1 and consists of one sink, i.e., the central control station, and n identical transmitting nodes, i.e., mobile robots and human workers. These are randomly distributed in a field of 50 m x 50 m and periodically transmit their position and speed to the sink located in the center of the field. All data is processed by the OMNeT++ framework at runtime, comparing time stamps of simulated packets to determine whether they overlap and, hence, interfere with each other. Again, for simplicity, external interference and clock drift are neglected for now, as this is discussed separately in Section 6.4.4 and 6.4.5. However, observations and conclusions remain valid.

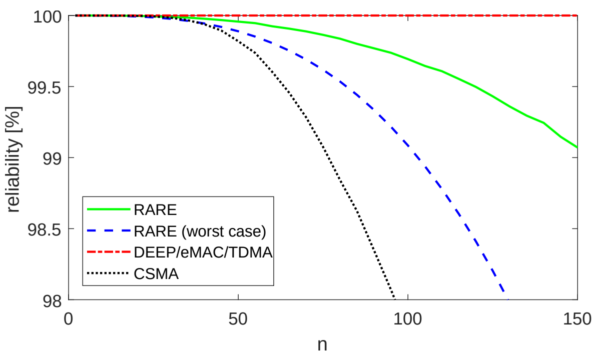


Figure 6.7.: Simulated reliabilities of the different MAC protocols for varying n. Note that the *"RARE (worst case)"*-curve shows the worst case computed by the formulas provided in Section 5.1.

Note that, in the case of RARE, network parameters were slightly changed from the previous numerical analysis to match the different nature of the Omnett++ simulation environment. That is, the packet numbers were reduced to $k = 3$ to reduce reliability and be able to show results in a clearer manner — for very high reliabilities, simulation results become inaccurate. However, this assumption does not invalidate the following results and $k = 3$ is sufficient for a good average performance. Lastly, based on the findings of the numerical analysis in Section 6.3.2, the parameters m and t_{min} were set to $m = 1$ and $t_{min} = \frac{t_{max}}{2}$ to maximize reliability.

6.4.1. Reliability and packet loss

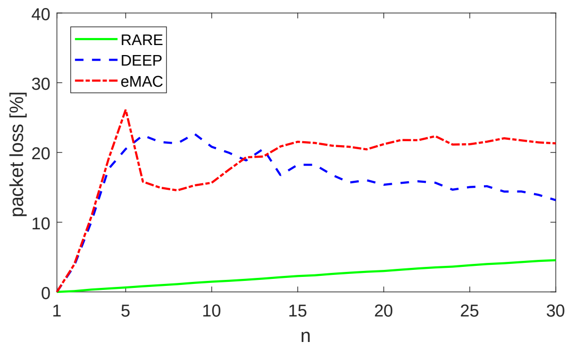
First, Fig. 6.7 displays the simulated (average) reliabilities of the different MAC protocols for rising n. It can be observed that DEEP, eMAC and TDMA offer an average reliability of 100 % for all n, i.e., there was never any data loss.[3] In the case of TDMA, this is because collisions between nodes are effectively prevented by its time slot arbitration, whereas the eMAC and DEEP protocols prevent data loss by creating unique, redundant transmission patterns.

In contrast, the reliability of CSMA and RARE decreases with a rising n, since more traffic is generated leading to higher collision probabilities. In the case of CSMA, reliability is the lowest due to its backoff mechanism that is designed for fast data transfer rather than reliability. In particular, backoff times are initially very short and only increased if retransmissions fail. Short backoff times cause packets to be sent faster and, therefore, lead to a higher channel utilization, i.e., a big amount of data per time unit. This generally increases collision probability, in particular, if multiple nodes are triggered simultaneously [9]. RARE, on the other hand, balances network load by distributing packets uniformly along the full deadline. This avoids peaks in data traffic and leads to a higher reliability, e.g., more than 99 % for $n \leq 150$ on average. Note that the average reliability of RARE never falls below its calculated lower bound (*"RARE (worst case)"*-curve), which validates theory.

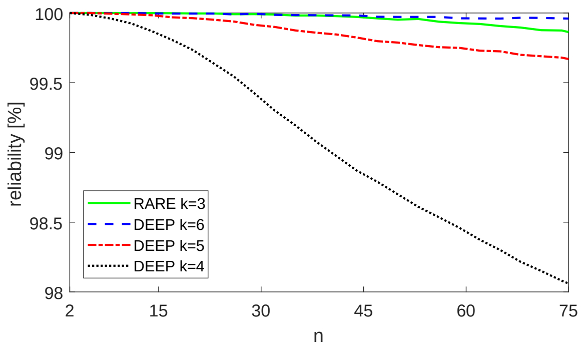
Next, Fig. 6.8a shows how many packets are lost on average within a packet sequence as n increases, which illustrates the amount of unnecessary redundancy of the unidirectional protocols.[4] A lower packet loss means that more packets from a sequence reach their destination, i.e., sending less packets would have been sufficient to guarantee relia-

[3]Note that eMAC could only be simulated up to $n = 30$, since finding inter-packet times is difficult due to its high computational complexity.

[4]Note that the bidirectional protocols cannot be included in Fig. 6.8a, since these do not transmit a fixed number of packets per sequences, but instead stop after receiving an ACK.



(a) Packet loss per sequence of the unidirectional protocols



(b) Simulated reliabilities for DEEP and RARE for different k and n

Figure 6.8.: Simulated packet loss and reliability

bility in this case. However, Fig. 6.8a shows the average case, whereas these protocols were designed to guarantee reliability in the worst case.

The amount of packet loss depends on the network size n, number of packets per sequence k and the time between transmissions (inter-packet times). Starting from 0 % for $n = 1$ — there are no collisions with only one node in the network — packet loss rises for all protocols with increasing n, since more collisions occur due to higher network traffic. In the case of RARE, which always transmits $k = 3$ packets per sequence, packet loss rises linearly. In addition, since it transmits less packets than eMAC or DEEP and distributes them more evenly along the available time (i.e., the deadline), it generally loses less packets per sequence.

In the case of DEEP and eMAC, a different behavior can be observed. As we know, the number of packets k per sequence equals the network size n, hence, increasing n leads to a strongly rising data traffic. On the other hand, inter-packet times also increase for higher n and, consequently, the amount of packet loss reduces. In this case, the effect of longer inter-packet times dominates over that of increasing packet numbers from a certain point onwards. This point is reached at $n = 4$, where the maximum packet loss can be observed. Note that the fluctuations in the curves of eMAC and DEEP are caused by their non-linear inter-packet times. Further, eMAC has a typically higher packet loss, due to is more optimized and shorter inter-packet times.

Fig. 6.8b evaluates how well DEEP performs with a reduced number of packets per sequence k compared to RARE. As can be seen here, when reducing k to a value smaller than n, DEEP cannot guarantee full reliability anymore, but starts losing data. This data loss increases with rising n, since more traffic is generated and more collisions occur, consequently leading to a lower reliability. Clearly, the lower the number of k, the faster reliability decreases, since it is more likely to lose all packets of a sequence. In this example, for $n = 75$, DEEP drops to a reliability of around 98 % for $k = 4$, 99.8 % for $k = 5$ and 99.99 % for $k = 6$. Note that these reliabilities are relatively high, considering that the packet numbers are much smaller than the $k = 75$ needed for full reliability. This again shows that full reliability is expensive, e.g., to get from 99.99 % ($k = 6$) to 100 % ($k = 75$) reliability, 69 more packets are required per sequence.

Note that RARE generally achieves a higher reliability than DEEP for the same packet numbers, e.g., it can achieve a reliability of 99.9 % for $n = 75$ and $k = 3$, which is higher than for DEEP with $k = 5$. This is because nodes transmit data in random time intervals that are independent of each other. With DEEP, on the other hand, the inter-packet times of all nodes depend on each other, which is very restricting. Although this has benefits such as allowing both fully-reliable and reduced-reliable nodes in the same network, this also slightly decreases performance compared to RARE.

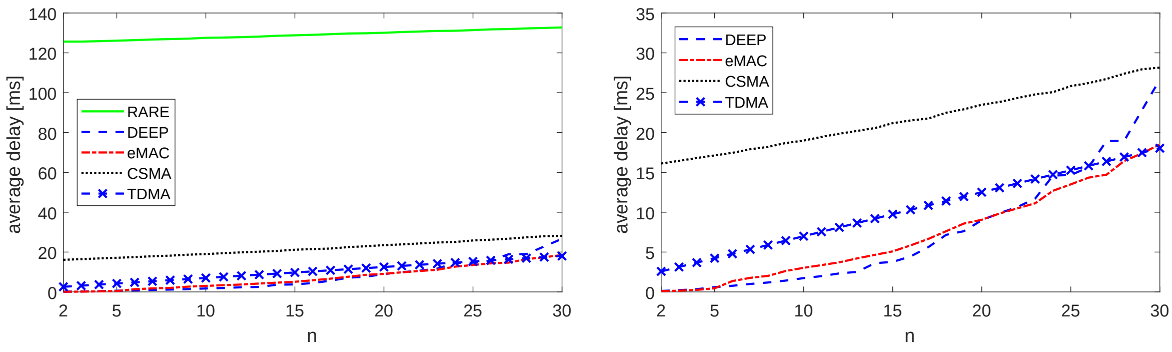


Figure 6.9.: Average delays of the different MAC protocols for a varying n. The right figure shows an enlarged view for better visibility.

6.4.2. Delay

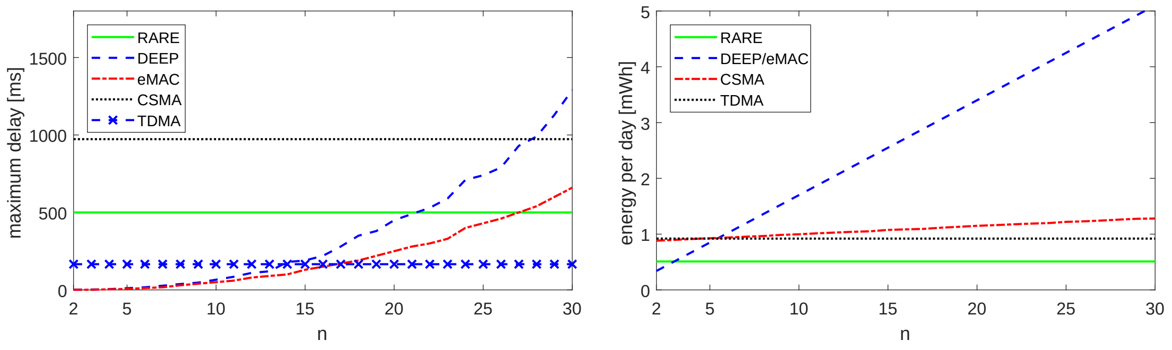
This section evaluates the communication delay of the different MAC protocols, i.e., the time needed from the activation of a node until the first successful reception of a packet. As can be seen in the case of the average delay in Fig. 6.9, an increasing n leads to longer average delays for all MAC protocols due to higher network traffic and increased collision probabilities. As a consequence, each node has to sent more packets on average for a successful reception.

In the case of DEEP and eMAC, the average delay rises quadratically as a result of longer inter-packet times for a greater n — this increase is less steep for eMAC due to its more optimized and shorter inter-packet times. TDMA, on the other hand, shows a linearly rising average delay due to a higher number of slots within the cycle. Similarly, CSMA and RARE also show linearly rising delays, however, the overall value is much higher for RARE. This is because, unlike *CSMA*, which tries to achieve fast transmission by first using short back-off times, RARE equally distributes its k transmissions within the full deadline d_{max}. This results in longer average delays, however, it also leads to a more balanced traffic load. For example, in bursty networks, e.g., when an event triggers multiple nodes at the same time, RARE yields no traffic peaks, resulting in less collisions in total. This is beneficial for reliability as shown previously in Fig. 6.7 and energy efficiency as shown in the next section.

The maximum delay, i.e., the longest time from activation of a node until successful reception of its data, is shown in Fig. 6.10a. Here, both RARE and TDMA offer constant maximum delays, which are independent of n. In the case of RARE, this delay is equal to the deadline d_{max}, whereas for TDMA this has the value of one full TDMA cycle, i.e., 167 ms. In contrast, CSMA, eMAC and DEEP have exponentially rising delays. For CSMA, this rises with the number of back-off retries and transmission attempts, hence, it is independent of n and shows a constant value in Fig. 6.10a. For eMAC and DEEP, however, the maximum delay increases exponentially with n^3 due to larger inter-packet times and packet numbers. This results in very high delays for large n, for example, 1.3 s for $n = 30$ for DEEP and 0.7 s for $n = 30$ for eMAC; note that eMAC features shorter delays due to more optimized and shorter inter-packet times. As a result, both protocols are only suitable for smaller networks or for applications that tolerate long delays.

6.4.3. Energy consumption

This section further analyzes the energy consumption by the different MAC protocols. Towards this, Fig. 6.10b depicts the average energy consumption per node on one day, in which the system was first active for 12 h and then in sleep for the remaining 12 h. The amount of energy consumed was calculated by multiplying the recorded times each node spent in receive, transmit and sleep mode with the corresponding power levels from the CC2545



(a) Maximum delays of the different MAC protocols (b) Average energy consumption per node for one day

Figure 6.10.: Maximum delays and energy consumption for varying n

transceiver [83]. That is, a node requires $62.5\,mW$ during transceiver switches and in receive mode, $80.5\,mW$ in transmit mode and $4.5\,\mu W$ in sleep mode at $V_{DD} = 3\,V$. Note that during the 12 h sleep period, TDMA is deactivated and no beacons are transmitted to save energy.

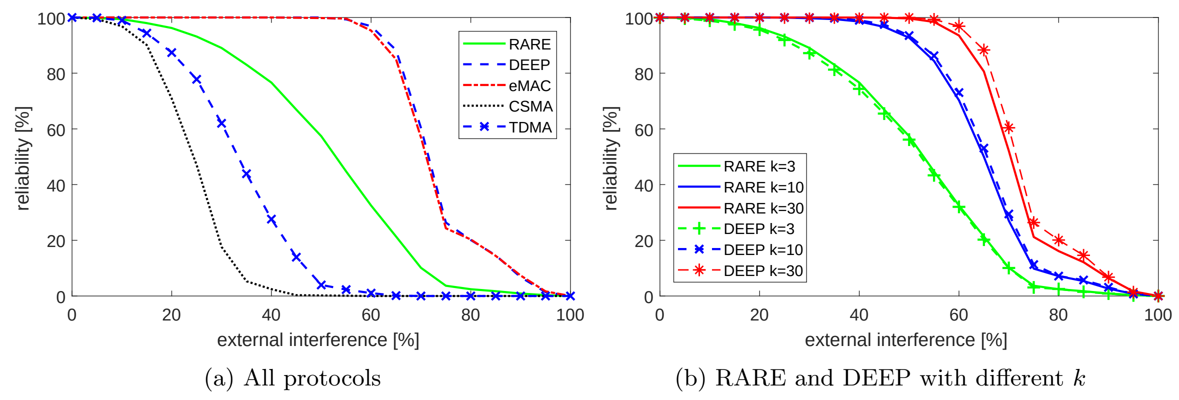
The energy consumption of a node depends on the number of packets sent, as well as on the packets themselves. For the unidirectional protocols eMAC, DEEP and RARE, a single packet transmission causes a node to be in transmit mode for $l_{max} = 88\,\mu$s, which requires $7\,\mu$Ws energy. When sending ACKs, a node has to switch to receive mode after sending a packet, listen for the ACK and switch back to transmit mode. Considering that each node switch takes $130\,\mu$s and receiving the ACK takes $56\,\mu$s, this results in additional $316\,\mu$s time in receive mode and a total energy consumption of $26.8\,\mu$Ws. With CSMA, nodes have to spend additional $150\,\mu$s in receive mode for carrier sensing, increasing the total receive time to $466\,\mu$s and energy consumption to $36\,\mu$Ws.

As shown in Fig. 6.10b, energy consumption of RARE is independent of n, since nodes always transmit three packets per deadline. Similarly, with TDMA, nodes always send one packet per deadline, since there are no (internal) collisions and external interference is neglected. However, since transmitting one packet with ACK requires more energy than transmitting three packets without ACK, TDMA consumes more energy than RARE in total. In addition, nodes have to periodically receive beacons, which makes up for around 18 % of the total energy demand. CSMA, on the other hand, shows an increasing energy consumption for rising n, as more collisions occur and packets have to be retransmitted more often — this first rises linearly until starting to saturate for very high n due to limited retransmission numbers. Lastly, in the case of eMAC and DEEP, in which nodes have to transmit $k = n$ packets per sequence, energy consumption also rises linearly, however, to much higher levels for large n.

6.4.4. External interference

So far, we considered internal interference only, i.e., packet collisions are caused by simultaneous transmissions within the network. In real-world deployments, however, there will inevitably be interference from sources outside the network, also referred to as noise on the communication channel. This noise typically originates from neighboring networks, for example, devices using WiFi, Bluetooth, etc., or other sources generating electromagnetic emissions, e.g., sparks in a rotating electric motor — see also Section 2.2.2.

To show possible effects on the different MAC protocols, a network of $n = 30$ nodes was simulated in which a separate node randomly emits interference pulses of variable length from 48 to $304\,\mu$s, which corresponds to the duration of data packets with payloads between 0 and 64 bytes. The level of interference, i.e., σ, can be changed by varying the duty cycle



(a) All protocols (b) RARE and DEEP with different k

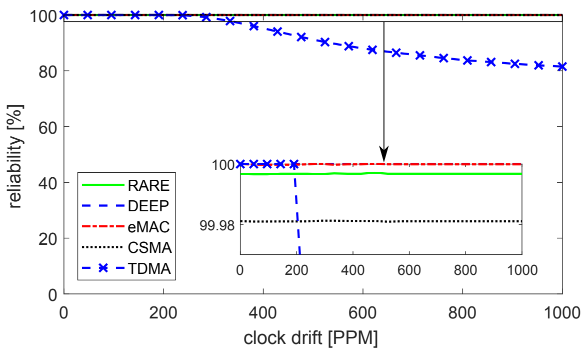
Figure 6.11.: Simulated reliabilities for an increasing level of external interference

of the interfering node, i.e., the ratio of time it actively transmits to the inter-pulse separation. This is stepwise increased from 0 % (no interference) to 100 % (blocked channel) in the following experiments. Finally, it is again (pessimistically) assumed that any overlapping of transmissions leads to packet loss, i.e., there is no capture effect and data cannot be recovered.

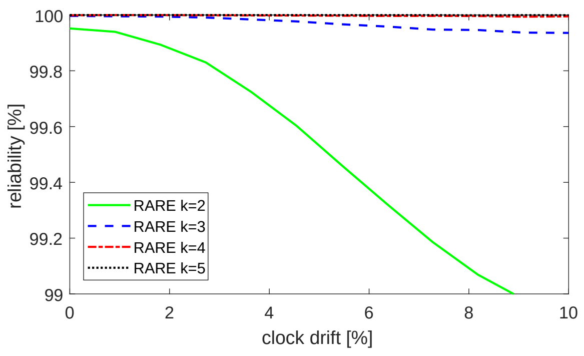
As can be seen in Fig. 6.11a, a rising level of interference decreases the reliability of all MAC protocols. This is because transmissions are corrupted more often and nodes have to perform a larger number of backoffs/retransmissions, leading to a higher chance that data can no longer be transmitted to the sink within the protocols' maximum retransmission numbers, etc. For low interference levels, reliability generally decreases only slightly, since each protocol can tolerate a certain amount of additional collisions/corruption. For higher levels of external interference, however, for example 10 % for CSMA, transmission numbers start to not be sufficient anymore and reliability decreases very quickly. For even higher levels of external interference, e.g., 35 % for CSMA, communication is hardly possible anymore and reliability converges to 0 %.

In general, bidirectional protocols are less robust against external interference compared to unidirectional protocols due to their higher protocol overhead, e.g., acknowledgments, carrier sensing and synchronization. These mechanisms are also affected by external interference and, hence, form additional error sources that lower reliability. In Fig. 6.11a, CSMA is the least robust against external interference, since it has the highest internal collision rate due to its backoff scheme, which is optimized for short delays rather than for reliability. As a result, there is less room for tolerating additional collisions/corruption compared to other protocols. In the case of TDMA, reliability is the second lowest due to the overhead created by its synchronization and ACK scheme. In particular, if too many synchronization beacons are lost, the network desynchronizes and communication is not possible anymore. For CSMA and TDMA, external interference should be below 10 and 15 % to still maintain reliabilities > 95 %.

Unidirectional protocols, on the other hand, are generally robust against external interference due to their low overhead. For example, RARE, shows a relatively high robustness, although it transmits only $k = 3$ packets per sequence, which is less than CSMA with up to 4 retransmissions and 3 backoff retries. As can be seen in Fig. 6.11a, it offers a reliability over 95 % for up to 20 % interference. DEEP and eMAC also show an excellent robustness against external interference, offering reliabilities of > 95 % for up to 55 % interference. This is due to their high transmission numbers ($k = 30$), which result in a very low probabilities that all packets of a sequence are corrupted. Note that eMAC and DEEP show very similar performance, since both transmit the same number of packets. As discussed next, this is the main factor of how robust a protocol is against external interference.



(a) All protocols for small clock drifts



(b) RARE for very large clock drifts

Figure 6.12.: Simulated reliabilities for an increasing clock drift

Next, Fig. 6.11b depicts the reliability of RARE and DEEP for an increasing level of interference and different numbers of packets per sequence. As mentioned before, the robustness of a protocol is largely determined by the number of packets per sequence — and by the packet length and overhead, which, however, are the same in this case. This is also shown in Fig. 6.11b, where it can be observed that both DEEP and RARE feature a very similar performance for the same k. The slight difference between both curves originates from their internal collision rates; the lower this value is, the higher the reliability against external interference. For small k, e.g., $k = 3$, RARE has a smaller internal collision rate than DEEP due to larger inter-packet times, which, however, reverses for larger k. This can also observed in Fig. 6.11b, where RARE performs slightly better for $k = 3$ and slightly worse for $k = 10$ and $k = 30$.

In summary, both RARE and DEEP feature an excellent robustness against external interference, clearly outperforming existing approaches, in particular, bidirectional ones. For example, at 50 % external interference, both CSMA and TDMA cannot convey data anymore, whereas RARE and DEEP (at default configuration) still provide a reliability of 60 % and 99.9 % respectively. This robustness can be adjusted, for example, more packets can be sent per sequence, if an environment is noisy. On the other hand, if there is not much external interference, e.g., in a well-shielded, indoor environment, packet numbers can be lowered to save energy.

6.4.5. Clock drift

The following section discusses the effects of clock drift, i.e., how the naturally occurring variance of an oscillator's frequency affects a node's behavior. To this end, an exemplary network with $n = 30$ nodes was simulated in which each node runs on a local clock with a random drift in $[-\Delta_{drift}, +\Delta_{drift}]$. Here, Δ_{drift} is the maximum drift of the current iteration, which is slowly increased from zero to 1000 ppm (= 0.1 %). Note that a drift of 1000 ppm means that the clock deviates 1000 parts per million from its nominal frequency. For example, if we consider a 1 MHz crystal, a 1000 ppm drift would cause it to run at either 1,001,000 Hz or 999,000 Hz.

In Fig. 6.12a, it can be observed that all asynchronous approaches, i.e., CSMA, eMAC, DEEP and RARE, are barely affected by an increasing clock drift due their lack of synchronization and repeated number of (re-)transmissions. In the case of CSMA, no decrease in reliability was measured, since inter-packet times are random, making the protocol robust against clock drift. With DEEP and eMAC, on the other hand, where inter-packet times are constant and carefully selected to generate collision-free transmission pattern, a clock drift $> 0\,ppm$ leads to data loss. That is, their transmission patterns are altered and, hence, cannot prevent multiple collisions between nodes anymore. However, the observed decrease in reliability is very small due to high packet numbers, i.e., it is still unlikely to lose all packets of a sequence. In this example, data loss was below 10^{-6} for all drift values $> 0\,ppm$.

In contrast, the synchronous TDMA protocol is strongly affected by clock drift due to its slot-based structure. That is, if drift values exceed a certain threshold — determined by the guard time interval and beacon rate — slot boundaries are violated and collisions occur, in particular, between neighboring slots. Beacons can also be corrupted by drifted packets, delaying resynchronization and, therefore, amplifying these effects even further. In Fig. 6.12a, it can be observed that data loss starts from $200\,ppm$ onwards. This is the value from which nodes can drift more than $t_{guard} = 700\,\mu$s before resynchronizing, i.e., before receiving another beacon every 3,5 s. The larger the drift values the stronger these effects are, i.e., it is more likely that a node drifts beyond its slot boundaries. For very high drift values, e.g., $1000\,ppm$, reliability saturates to around 80 %.

In the case of RARE, reliability also decreases for rising clock drifts, however, only very slightly (hard to see in Fig. 6.12a). This is caused by a superposition of the following effects: First, when the clock drift of a node is negative, i.e., the clock frequency is lower, it counts time at a slower rate. This will enlarge the interval $t_{min} - t_{max}$ and therefore increase reliability. However, since t_{max} also increases, packets of a node can now miss the deadline as (5.1.6) is violated. Second, if the clock drift of a node is positive, it counts time at a higher rate, which results in a shorter interval $t_{min} - t_{max}$ and, hence, a decreased reliability. Additionally, (5.1.1) can start being violated and therefore reliability decreases further. In summary, it can be observed that negative effects dominate on average and reliability decreases for a higher clock drift. This is depicted more clearly in Fig. 6.12b, where clock drifts up to 10 % (equal to $100,000\,ppm$) have been simulated. Here, it can also be observed that a higher k is generally more robust, since it is less probable to lose all packets of a sequence.

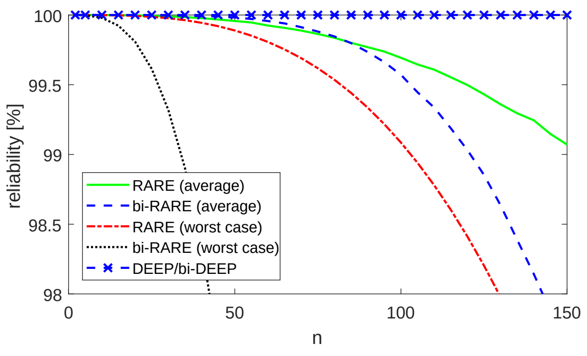
Note that the above experiments show very high clock drift values and in real-life, these will typically be much smaller. For example, even cheap crystal oscillators can guarantee a 100 ppm = 0.01 % accuracy or better. For this value, the simulated reliability in Fig. 6.12b deteriorates by less than 0.02 % for $k = 2$ and 0.001 % for $k = 3$. For $k \geq 4$ the change is too small to be measured. Thus, in most of the cases, clock drift can safely be neglected when designing a network with RARE, i.e., it is not necessary to use (5.4.1) and (5.4.1). Similarly, with DEEP and eMAC, where packet numbers are even higher and the measured data loss much smaller ($< 10^{-6}$), clock drift can also be safely neglected in most cases.

6.4.6. Bi- vs unidirectional communication

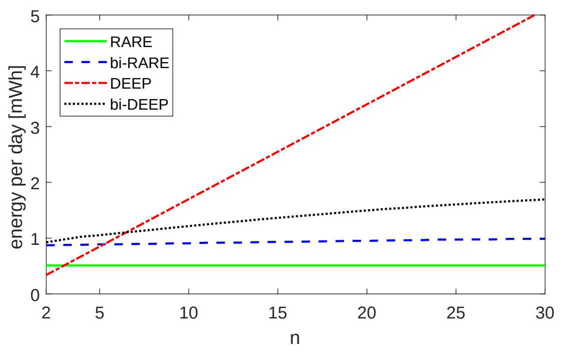
In this section, the bidirectional protocols bi-DEEP and bi-RARE are further evaluated and compared to their unidirectional counterparts DEEP and RARE. To this end, the following analysis first starts with evaluating the reliability of the different protocols.

Fig. 6.13a displays the simulated (average) and calculated (worst-case) reliabilities of each protocol. As can be observed, there is no difference in the average and worst-case performance of DEEP and bi-DEEP and both protocols achieve 100 % reliability. RARE and bi-RARE, on the other hand, show different behavior in both cases. That is, in the worst case, using an ACK scheme greatly deteriorates the reliability by introducing two additional error sources: the ACK packet itself and the transceiver switching times in which the sink can potentially miss packets. This deterioration is independent of k, meaning that the worst-case reliability with ACKs is always lower than without ACKs for the RARE-based protocols. Note that bi-DEEP accounts for this by selecting appropriate inter-packet times, hence, reliability is not affected.

In the case of the average reliabilities in Fig. 6.13a, bi-RARE first offers a slightly higher reliability for small n than RARE. That is, in small networks, the data traffic is low enough that ACKs prevent more collisions than they create. For higher n, however, this effect reverses

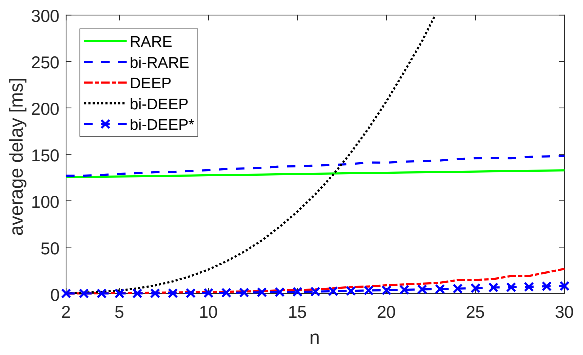


(a) Reliabilities

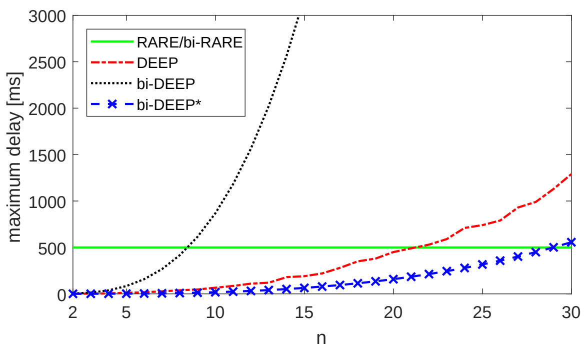


(b) Daily energy consumption of a node

Figure 6.13.: Simulated (average) and calculated (worst-case) reliabilities and the daily energy consumption for rising n



(a) Average delay



(b) Maximum delay

Figure 6.14.: The average and maximum delay for an increasing n. Note that bi-DEEP* is an optimized variant of bi-DEEP using a faster transceiver.

as the traffic load increases and the channel starts to saturate. In the above example, bi-RARE offers a lower reliability from $n = 80$ onwards than RARE. The same effect can be observed when varying other parameters, such as the deadline, packet lengths, etc. In general, the higher the channel load, the more ACKs become a burden than a help, hence, the unidirectional RARE performs better. For more details on this, see also [67].

Next, Fig. 6.14a shows the average communication delays of the different MAC protocols, i.e., the time needed from the activation of a node until the first successful reception of a packet. As expected, the average delays increase for rising n for all protocols, since more packets need to be transmitted on average before one is received successfully. In the case of RARE and bi-RARE, the delay rises slowly in a linear fashion, whereby bi-RARE shows a slightly higher delay due to its ACK and carrier sensing scheme; these increase the active time of a node and, therefore, lead to a higher chance that transmissions are not successful — see also Appendix B. For small n, i.e., at low traffic, this difference is very small and RARE and bi-RARE roughly have the same average delay. For larger n, however, this difference increases. The maximum delay, on the other hand, is the same $d_{max} = 500\,\text{ms}$ for both protocols, as shown in Fig. 6.14b.

In the case of DEEP and bi-DEEP, both the average and maximum delays rise with n^3 due to higher packet numbers and larger inter-packet times. While these are still relatively low for DEEP, bi-DEEP, on the other hand, shows very large delays. For example, for $n = 30$, bi-DEEP requires 0.6 s on average and 27 s in the worst case for a data transfer, whereas DEEP only requires 26 ms and 1.3 s respectively. These large differences result from their different methods to calculate inter-packet times. In the case of DEEP, these mainly depend on the packet lengths, whereas, with bi-DEEP, these depend on the chosen transceiver IC, i.e., on its sensitivity and switching speed. Since, in this example, packet lengths ($l_{max} = 88\,\mu$s) are relatively short compared to the transceiver switching time ($t_{set} = 130\,\mu$s) of the CC2545 [83], DEEP offers shorter delays. However, if a faster transceiver is used instead, for example, the MAX2837 [56] with $t_{set} = 2\,\mu$s, the delays of bi-DEEP can be reduced considerably as shown by the "bi-DEEP*"-curves in Fig. 6.14a and Fig. 6.14b. Data transfers then only require 8 ms on average and 0.5 s in the worst case for $n = 30$.

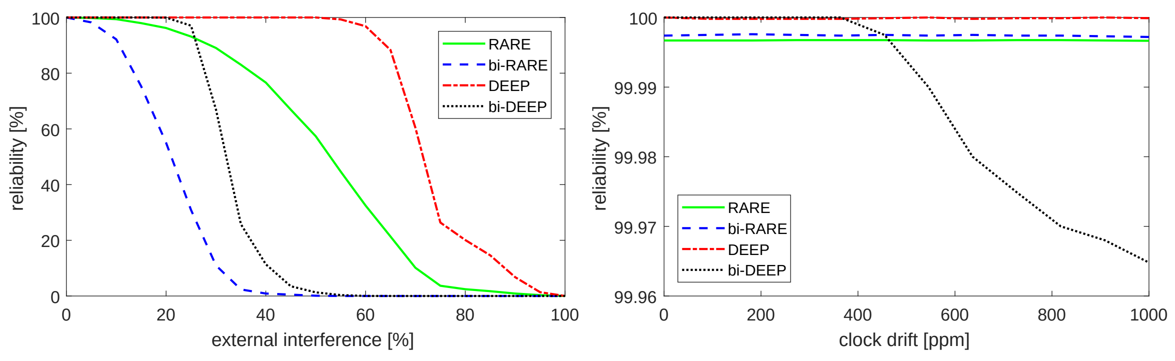
Although it seems an obvious choice to use only fast transceivers for bi-DEEP, this also has some disadvantages. To understand these, recall that the speed of a transceiver, i.e., its switching times, mainly depend on its circuit speed and modulation [73], which is OFDM (Orthogonal Frequency-Division Multiplexing) for the MAX2837 and GFSK (Gaussian Frequency-Shift Keying) for the CC2545. In general, different types of modulation have different advantages and disadvantages and might not be well suited for all applications. For example, OFDM typically incurs in higher vulnerability to external interference and increased costs [77]. As a result, if bi-DEEP is required to have short delays, the disadvantages of OFDM that have to be accepted in return. Lastly, note that the fluctuations in the curves of DEEP are caused by its nonlinear inter-packet times — bi-DEEP, on the other hand, has no fluctuations, since it is based on an analytic model instead of a heuristic one.

Next, Fig. 6.13b displays the average energy consumed by the different protocols. Towards this, similar as before in Section 6.4.3, a system of $n = 30$ nodes was considered that was first active for 12 h and then in sleep for the remaining 12 h. The amount of energy consumed was calculated by multiplying the recorded times each node spent in receive, transmit and sleep mode with the corresponding power levels from the CC2545 transceiver [83]. That is, a node requires $62.5\,mW$ during transceiver switches and in receive mode, $80.5\,mW$ in transmit mode and $4.5\,\mu W$ in sleep mode at $V_{DD} = 3\,V$.

As can be observed, the bidirectional protocols show a different behavior than their unidirectional equivalents. In particular, their energy consumption is not constant, but it increases with rising n, since more collisions occur and more packets have to be transmitted on average. While this increase is very small for bi-RARE, it is more pronounced for bi-DEEP, since more backoffs/collisions occur on average; this is due to differences in inter-packet times, which, in this example, are a bit shorter for bi-DEEP.

In general, the bidirectional protocols have smaller collision numbers, since carrier sensing prevents interrupting ongoing transmissions and ACKs reduce the number of packets sent on average. This positively affects energy consumption, in particular, for the bi-DEEP protocol, which offers a (much) lower energy consumption for $n > 6$ compared to DEEP. For smaller n (and k), however, the overhead of carrier sensing and ACK outweigh their benefits and energy consumption is higher. In this example, sending one packet with ACK and carrier sensing requires more than three times as much energy compared to a single unidirectional transmission — see also Section 6.4.3. As a consequence, bi-RARE always has a higher energy consumption compared to RARE, although it only transmits one or two packets per sequence on average.

Next, Fig. 6.15a depicts the effects of external interference on reliability of the different protocols — the same network as before in Section 6.4.4 has been simulated. As expected, reliability decreases for a rising level of interference, since packets are corrupted more often, making it more likely that data cannot be conveyed within the protocols' maximum (re-)transmission numbers anymore. Again, the bidirectional protocols show a lower overall performance, since carrier sensing and ACK mechanisms are affected by external interference as well creating additional errors. In the case of bi-DEEP, reliability is higher than for bi-RARE due to a larger number of retransmissions, i.e., $k = 2n - 1$ instead of $k = 3$. Similarly, for interference below 30 %, reliability is also higher than for RARE, which, however, then



(a) External interference

(b) Clock drift

Figure 6.15.: The simulated reliabilities of the different MAC protocols for varying levels of external interference and clock drift.

reverses, as the low overhead of unidirectional communication starts to dominate over the increased transmission numbers. Lastly, DEEP again shows the highest robustness against external interference due to its low overhead and large number of transmissions.

Fig. 6.15b shows the effects of clock drift on reliability of the different protocols — the same network as before in Section 6.4.5 has been simulated. It can be observed that bi-RARE shows a very similar behavior to RARE and reliability decreases very slightly for higher drift values. In this case, the additional carrier sensing and acknowledgment mechanisms of bi-RARE do not affect its robustness against clock drift. The bi-DEEP protocol, on the other hand, shows a different behavior compared to DEEP. That is, its performance decreases noticeable for larger clock drifts, e.g., to 99,95 % $(1-5\cdot 10^{-4})$ for 1000 *ppm* compared to only $(1-10^{-6})$. This lower robustness is mainly caused by its different inter-packet times that are more sensitive to distortion; these are relatively short with respect to a full transmission with carrier sensing and ACK, which, together with the higher complexity, makes bi-DEEP more vulnerable to data loss. However, note again that a major decrease in performance can only be observed for relatively large clock drifts $> 350\,ppm$, whereas for smaller drifts, data loss was similarly low as for DEEP, e.g., 10^{-6}. As a result, since even cheap crystal oscillators have an accuracy of 100 *ppm* or better, clock drift can be safely neglected in most cases.

6.5. Summary

This chapter evaluated the proposed MAC protocols DEEP and RARE and compared them to state-of-the-art solutions from the literature. To this end, an exemplary case study was introduced first to demonstrate the design of a WSN and to obtain suitable values for the following experiments. Next, the compared protocols were introduced and two sets of experiments were performed: a numerical analysis and a simulation-based evaluation.

The numerical analysis evaluated DEEP and RARE with respect to their worst-case performance. To this end, both protocols were tested for a series of different network and MAC parameters and the results were recorded and evaluated. This illustrated the complex dependencies between different parameters and the corresponding network behavior. Finally, it was analyzed how well DEEP and RARE perform in heterogeneous networks, i.e., with arbitrary node types having different deadlines and packets lengths. Results showed that both protocols greatly benefit from different node types allowing for large improvements and, in the case of DEEP, even outperforming the more elaborate eMAC protocol. Lastly, it was shown that DEEP can greatly reduce computational complexity compared to eMAC and that

RARE can achieve high reliabilities even for low packet numbers. For example, in a network of 75 nodes, RARE achieves a reliability of 99.6 % with only 3 packets per sequence.

The second part covered a simulation-based evaluation examining the average behavior of DEEP and RARE and comparing them to state-of-the-art protocols from the literature. Results are presented in Table 6.1. It was observed that DEEP and eMAC both achieved 100 % reliability in simulation, i.e., there was no data loss recorded. However, they generally suffered from high energy consumption and long delays, due to large packet numbers and inter-packet times — both increase with the network size n. Their robustness against clock drift and external interference, on the other hand, was excellent due to high packet numbers. Note that eMAC has a lower complexity rating in Table 6.1 than DEEP, since calculating inter-packet times is computationally expensive.

Property	**DEEP**	**bi-DEEP**	**RARE**	**bi-RARE**	**eMAC**	**CSMA**	**TDMA**
Unidirectional	✓		✓		✓		
Reliability	+++	+++	++	++	+++	+	+++
Average delay	++	+	+	+	++	++	+++
Maximum delay	-	-	++	++	-	+	+++
Energy consumption	+	++	+++	++	+	++	++
Ext. interference	+++	+	++	+	+++	+	+
Clock drift	+++	++	+++	+++	+++	+++	+
Complexity	+++	++	+++	++	++	++	-
Overhead	++	+	+++	+	++	+	-

Table 6.1.: The properties of the different protocols. '-' means poor, '+' medium, '++' good and '+++' excellent.

RARE achieved very high performance in all categories, except for the average delay. This is higher than for the other protocols, since RARE distributes its transmissions randomly over the full deadline rather than transmitting them as fast as possible; as shown for CSMA, transmitting as fast as possible reduces delays, but strongly increases collision numbers in return. Lastly, the average simulated reliability of RARE never fell below its calculated worst case one, which validates its theory.

The protocols bi-DEEP and bi-RARE showed mixed behavior, i.e., compared to DEEP and RARE, some features have been improved, others worsened. In the case of bi-RARE, performance was similar to RARE, but nodes required a bit more energy and incurred in higher overhead and complexity due to additional carrier sensing and ACK mechanisms. However, since these mechanisms generally depend on network parameters such as packet numbers k, transceiver speed, etc., results can be different for other applications/settings. In general, bidirectional protocols perform better for high packet numbers, fast transceiver with short switching times, and low traffic — under opposite conditions, e.g., very high traffic, ACKs turned out to be more a burden than of help. In the case of bi-DEEP, reliability was always 100 % and energy consumption was improved compared to DEEP, since less packets were sent on average. However, all other performance characteristics, especially delays, are greatly impaired by a larger overhead — again, a faster transceiver can be used to increase performance, which, however, leads to higher costs.

CSMA showed moderate performance, but particularly suffered from a low reliability due to its ineffective backoff scheme designed for short delays rather than reliability. In the presented experiments, which simulated high network load for which CSMA is not designed, many packet collisions occurred and nodes dropped out quickly as they reached their maximum retransmission/backoff numbers. Similar behavior was observed for external interference, where CSMA performed poorly. Clearly, for lower traffic load (or less external interference) CSMA would perform better, but so would the other bidirectional protocols as well.

Lastly, the synchronous TDMA protocol achieved very low delays and a high reliability of 100 % without external interference. However, its slotted design is generally vulnerable against clock drift and external interference and it has a higher complexity and overhead due its synchronization mechanism.

In summary, each of the compared MAC protocols has its own strengths and weaknesses, making it be best suited for a specific applications. An overview of such exemplary application areas for TDMA and CSMA can be found in [74] [97] and for eMAC in [10] [11]. In the case of DEEP and RARE, these are discussed in the next chapter together with further protocol properties.

Chapter 7. Concluding remarks

This work presented DEEP and RARE, two MAC protocols designed for highly reliable communication in unidirectional WSNs, i.e., an application area where most other protocols typically fail to provide any quality of service (QoS) guarantees. In the following, these protocols are shortly summarized and their strengths and weaknesses are discussed. In addition, their properties are illustrated and possible application areas mentioned. Lastly, this chapter concludes this work by providing a brief outlook to possible future extension and modifications to the proposed protocols.

The first presented protocol, DEEP, enables *fully* (100 %) reliable communication, i.e., there is no data loss within the network (without external interference). Towards this, nodes transmit data as sequences of redundant packets, whereby packet numbers and inter-packet times are carefully selected to generate robust transmission patterns. This way, it can guaranteed that at least one of the redundant packets reaches its destination in the worst case.

DEEP's main strength is its high robustness against both internal and external interference. During simulation, no data loss was recorded due to internal interference, even after simulating several million data sequences. Similarly, the measured robustness against external interference was exceptionally high. For example, in a very noisy environment with 55 % external interference, reliability was still above 99 %, whereas other protocols such as CSMA dropped below 1 %, i.e., no communication was possible anymore. Further, DEEP is easy to implement, i.e., nodes transmit in constant time intervals for which a simple timer is sufficient. And lastly, DEEP features a much lower computational complexity compared to similar approaches from the literature. For example, calculating inter-packet times for $n = 30$ nodes required 40 hours with eMAC [10], but only several ten seconds with DEEP. Although these computations normally run off-line on a fast computer, short computation times can facilitate testing and debugging as many different settings can be emulated in a short time.

Despite many advantages, DEEP also has several drawbacks. In particular, its ability to enable full reliability comes at the costs of increased packet numbers and long inter-packet times — these rise linearly and quadratically for larger network sizes n. As a result, to avoid high energy consumption and longs delays, DEEP should only be used for smaller networks, for example, home automation applications with less than $n = 50$ nodes [68]. Further, DEEP suffers from low flexibility regarding network dynamics, i.e., when nodes enter and leave the network, inter-packet times must be changed for each node. This can be difficult due to the limited reconfiguration possibilities of unidirectional nodes, however, more on this later.

The second presented protocol, RARE, is also designed for highly reliable communication in unidirectional WSNs. In contrast to DEEP, which guarantees full reliability, i.e., data always reaches its destination in the worst case, RARE makes use of the fact that many applications can tolerate some data loss. In other words, it can be configured to a user-specified reliability of less than 100 %, i.e., to a certain probability that data reaches its destination within a deadline in the worst case. Towards this, nodes send their data as sequences of k redundant packets with random inter-packet times selected from an interval $[t_{min}, t_{max}]$. By adjusting t_{min}, t_{max} and k, reliability can be adapted to the given requirements.

Tolerating packet losses significantly reduces pessimism compared to DEEP's deterministic model and allows using a probabilistic model. This has the advantage that it offers a lower complexity, i.e., performance and network parameters are easier to calculate, and avoids the high overhead of full reliability, i.e., the large packet numbers and long inter-packet times. For example, in a network of $n = 30$ nodes, DEEP requires each node to send $k = 30$ packet per sequence for full reliability. RARE, on the other hand, can achieve a reliability $p = 99,98\%$ with only $k = 3$ packets when assuming the parameters from Section 6.1. This is a 10-times reduction in energy consumption — energy is directly proportional to the number of packets sent — while the chance of losing data is only $0,02\,\%$ in the worst case.

However, RARE also has some drawbacks, especially, relatively high average delays. This is because RARE distributes its packets evenly over the full available time (i.e., deadline) rather than transmitting them as fast as possible as, for example, CSMA. While this avoids peaks in traffic and reduces collision numbers, it also leads to relatively high delays on average. However, note that many applications do not mind these delays as long data arrives within a maximum time bound or deadline. Another disadvantage of RARE is that it cannot guarantee very high reliabilities close to $100\,\%$ as packet numbers increase strongly due to its random architecture. In such cases, it is more meaningful to use DEEP instead.

Lastly, both DEEP and RARE provide several advantages and features that distinguish them from similar approaches from the literature. In particular, both protocols do not pursue best effort approaches, but provide frameworks to easily access network performance. To this end, their analyses include practical factors such as clock drift and external interference, and support heterogeneous networks, i.e., nodes with arbitrary packet lengths and deadlines. This has the advantage that packet loss probabilities can be modeled more accurately, which in return improves reliability and the maximum achievable network sizes — see the experiments in Section 6.3. And finally, both protocols can adjust their packet numbers individually for each node and, therefore, save energy whenever data loss can be tolerated.

7.1. Protocol properties and application areas

This section explains the characteristics of DEEP and RARE with respect to the five basic requirements of a WSN: reliability, complexity, energy consumption, scalability and latency — see also Chapter 1. Results are displayed in Fig. 7.1.

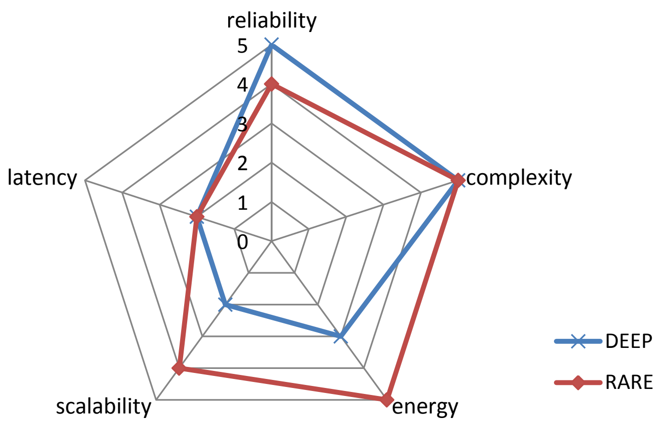


Figure 7.1.: The performance of DEEP and RARE with regard to the five basic requirements of a WSN (higher is better).

- **Reliability** describes the probability that data is successfully delivered within a specified time boundary. Since both protocols are explicitly designed for reliable communication, they show very high ratings: DEEP offers *fully* reliable communication between nodes in the network and RARE, which is designed to tolerate slight data loss, can be configured to a reliability $< 100\,\%$. In addition, both protocols show high robustness against external interference — see again Section 6.4.4.

- **Complexity** describes the costs of implementing and operating a network. Both DEEP and RARE feature a very low complexity due to their unidirectional design. That is, nodes do not implement elaborate mechanisms, but simply transmit data packets periodically.

- **Energy consumption** describes how well the network can economize on energy and, therefore, defines the lifetime of a battery-powered system. In the case of RARE, energy consumption is excellent due to short transmissions without carrier sensing or ACKs and typically low packet numbers. With DEEP, on the other hand, transmission numbers depend on the network size n, hence, energy is good in small networks, but decreases for larger n.

- **Scalability** describes how well the system can handle changing network sizes n, i.e., when nodes enter or leave dynamically. DEEP features a poor scalability, since inter-packet times depend on n, hence, nodes must change their transmission patterns every time a node enters or leaves the network. This can be particularly difficult in purely unidirectional networks, where nodes cannot receive notification or reconfiguration data — these then must be reprogrammed manually [68]. RARE, on the other hand, features good scalability, since inter-packet times are random and must not necessarily be adapted. Instead, the resulting performance can be easily calculated by the provided formulas — reliability increases when nodes leave the network and decreases when nodes join the network.

- **Latency** describes the time required to transfer data or, vice versa, how much data can be transfered in a given period of time (i.e., throughput). Both protocols are not designed for low latencies or high throughputs, but instead sacrifice this attribute to achieve good reliability. In this context, short (average) delays are not very important, as long as it can be ensured that data arrives within a maximum delay or deadline.

Note that further, more general protocol properties and characteristics can be found in Table 7.1 at the end of this chapter. This also includes information about the widely-used protocols CSMA and TDMA, i.e., bidirectional protocols designed for flexibility and high throughput respectively, and eMAC, a unidirectional protocol similar to DEEP. More details on the protocols can be found in Section 6.2.

Regarding possible application areas, both DEEP and RARE are generally well-suited for environmental monitoring applications, for example, forest fire detection [50] or climate reporting [49]. These are systems where nodes are idle for long times, e.g., for hours, days or weeks, but need to transport data in a timely manner when being activated. In addition, DEEP and RARE can also be used in wireless body area networks (WBANs) to improve costs and reliability. These are typically small networks consisting of heart rate sensors, pedometers, etc. that are attached to a person and periodically report data to a base station, e.g., a health monitoring application on a phone [46] [76] [91]. Another well-suited application is home- and building automation, where electronic devices are interconnected to enable more elaborate control or improve energy efficiency [44] [79] — see also Fig. 1.1 in Chapter 1. Or vehicular networks, in which electronic control units (ECUs) exchange data wirelessly [58] or communicate with an intelligent crossroad to efficiently control traffic flows [55]. The are many further application areas, such as IoT [60] [62], radio frequency identification (RFID) [37], smart grids [29], etc. In summary, DEEP and RARE are suitable for all applications that require reliable and timely data delivery at low costs, i.e., by using unidirectional communication. In this context, DEEP is suited for safety critical devices requiring very high reliabilities, for example, heating controllers or security devices in a home automation setting. RARE, on the other hand, is more eligible for applications that tolerate slight data loss, but require a lower energy consumption instead.

7.2. Outlook

To conclude this chapter (and this work), this section briefly discusses possible future extensions of DEEP and RARE that, for example, mitigate their disadvantages or add new features making them applicable for a wider range of applications. To this end, an interesting approach is to use hybrid networks, i.e., combine unidirectional nodes with bidirectional ones, to increase the flexibility of the network. For example, in mobile networks where nodes can join or leave the system at arbitrary points in time, network parameters must be reconfigured frequently. Bidirectional communication can be used for nodes with high dynamics to facilitate reconfiguration, whereas the less dynamic or static ones use unidirectional communication instead to save costs.

Another plan for the future is to improve the performance of DEEP in large networks, i.e., to reduce the high energy consumption and long communication delays. To this end, clustering can be used to virtually decrease the network size by dividing it into smaller sub-networks, i.e., clusters that are interconnected by cluster-heads [7] — see Section 3.2.2. Nodes within a cluster can then reduce their transmission power to avoid interference with neighboring clusters [89] or use different radio frequencies for communication [81]. As a result, if the system is set up correctly, only nodes within a cluster can interfere with each other, which are typically only a few nodes. This allows DEEP to use much lower network sizes for calculation and, thus, strongly reduce energy consumption and delays. Clearly, RARE also benefits from such methods, since these generally reduce interference within a network.

Property	**DEEP**	**RARE**	**eMAC**	**CSMA**	**TDMA**
Classification	asynchronous	asynchronous	asynchronous	asynchronous	synchronous
Unidirectional	✓	✓	✓		
Predictable performance	✓	✓	✓		✓
Robust against ext. interference	✓	✓	✓		
Fully reliable	✓		✓		✓
High energy efficiency		✓			
Strength	reliability	reliability	reliability	flexibility	throughput
Weakness	energy/delays	average delay	energy/delays	reliability	complexity
Reliability	+++	++	+++	+	+++
Complexity	+++	+++	++	++	-
Energy consumption	+	+++	+	++	++
Scalability	+	++	+	+++	+
Latency	-	+	-	+	+++

Table 7.1.: Further properties and characteristics of DEEP and RARE together with other commonly-used and/or similar protocols from the literature — see also Section 6.2. Note that '-' means poor, '+' medium, '++' good and '+++' excellent.

Bibliography

[1] Standard for Broadband Wireless Access Systems, IEEE 802.16-2017.

[2] Standard for Digital Enhanced Cordless Telecommunications (DECT), ETSI EN 300 175.

[3] Standard for Local and Metropolitan Area Networks, IEEE 802.11-2016.

[4] Standard for Low-Rate Wireless Networks, IEEE 802.15.4-2015.

[5] The OSI Model's Seven Layers Defined and Functions Explained. https://support.microsoft.com/kb/103884, Retrieved August 2018.

[6] A. T. Aby, A. Guitton, P. Lafourcade, and M. Misson. SLACK-MAC: Adaptive MAC Protocol for Low Duty-Cycle Wireless Sensor Networks. In *Proceedings of the Springer Ad Hoc Networks (Adhocnets)*, 2015.

[7] A. Ahmad and Z. Hanzalek. An Energy Efficient Schedule for IEEE 802.15.4/ZigBee Cluster Tree WSN with Multiple Collision Domains and Period Crossing Constraint. *IEEE Transactions on Industrial Informatics*, 14:12–23, 2018.

[8] I. Al-Anbagi, M. Erol-Kantarci, and H. T. Mouftah. A Survey on Cross-Layer Quality-of-Service Approaches in WSNs for Delay and Reliability-Aware Applications. *IEEE Communications Surveys Tutorials*, 18:525–552, 2016.

[9] G. Anastasi, M. Conti, and M. D. Francesco. A Comprehensive Analysis of the MAC Unreliability Problem in IEEE 802.15.4 Wireless Sensor Networks. *IEEE Transactions on Industrial Informatics*, 7:52–65, 2011.

[10] B. Andersson, N. Pereira, and E. Tovar. Delay-Bounded Medium Access for Unidirectional Wireless Links. In *Proceedings of the International Conference on Real-Time Networks and Systems (RTNS)*, 2007.

[11] B. Andersson, N. Pereira, and E. Tovar. Delay-Bounded Medium Access for Unidirectional Wireless Links. Technical report, CISTER - Research Centre in Real-Time and Embedded Computing Systems, 2007.

[12] Y. E. Aslan, I. Korpeoglu, and Ö. Ulusoy. A framework for use of wireless sensor networks in forest fire detection and monitoring. *Elsevier Journal of Computers, Environment and Urban Systems*, 36:614 – 625, 2012.

[13] M. S. Bahbahani and E. Alsusa. A Cooperative Clustering Protocol With Duty Cycling for Energy Harvesting Enabled Wireless Sensor Networks. *IEEE Transactions on Wireless Communications*, 17:101–111, 2018.

[14] B. Blaszczyszyn and B. Radunovic. Using Transmit-only Sensors to Reduce Deployment Cost of Wireless Sensor Networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[15] E. Borgia. The Internet of Things Vision: Key Features, Applications and Open Issues. *Elsevier Journal of Computer Communications*, 54:1 – 31, 2014.

[16] S. Brienza, M. Roveri, D. D. Guglielmo, and G. Anastasi. Just-in-Time Adaptive Algorithm for Optimal Parameter Setting in 802.15.4 WSNs. *ACM Transactions on Autonomous and Adaptive Systems*, 10:27, 2016.

[17] M. Buettner and D. Wetherall. A Software Radio-based UHF RFID Reader for PHY/MAC Experimentation. In *Proceedings of the IEEE Conference on RFID (IEEE RFID)*, 2011.

[18] S. Cao and V. C. S. Lee. A Novel Adaptive TDMA-Based MAC Protocol for VANETs. *IEEE Communications Letters*, 22(3):614–617, March 2018.

[19] R. Cardell-Oliver, A. Willig, C. Huebner, T. Buehring, and A. Monsalve. Error Control Strategies for Transmit-only Sensor Networks: a Case Study. In *Proceedings of the IEEE International Conference on Networks (ICON)*, 2012.

[20] P. Cardieri. Modeling Interference in Wireless Ad Hoc Networks. *IEEE Communications Surveys Tutorials*, 12:551–572, 2010.

[21] E. Celada-Funes, D. Alonso-Roman, C. Asensio-Marco, and B. Beferull-Lozano. A Reliable CSMA Protocol for High Performance Broadcast Communications in a WSN. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2014.

[22] X. Chang, J. Huang, S. Liu, G. Xing, H. Zhang, J. Wang, L. Huang, and Y. Zhuang. Accuracy-Aware Interference Modeling and Measurement in Wireless Sensor Networks. *IEEE Transactions on Mobile Computing*, 15:278–291, 2016.

[23] F. Chen and R. Li. Sink Node Placement Strategies for Wireless Sensor Networks. *Springer Journal on Wireless Personal Communications*, 68:303–319, 2013.

[24] W. Chen, C. T. Lea, S. He, and Z. XuanYuan. Opportunistic Routing and Scheduling for Wireless Networks. *IEEE Transactions on Wireless Communications*, 16:320–331, 2017.

[25] X. Chen and J. Zhou. Adaptive Packet Length Strategy to Minimize End-to-End Latency for Time-Varying Channels. In *International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, 2016.

[26] C.-Y. Chong and S. P. Kumar. Sensor Networks: Evolution, Opportunities, and Challenges. *Proceedings of the IEEE*, 91:1247–1256, 2003.

[27] B. Dezfouli, M. Radi, K. Whitehouse, S. A. Razak, and H.-P. Tan. CAMA: Efficient Modeling of the Capture Effect for Low Power Wireless Networks. *ACM Transactions on Sensor Networks*, 11:20, 2014.

[28] M. Doudou, D. Djenouri, and N. Badache. Survey on Latency Issues of Asynchronous MAC Protocols in Delay-Sensitive Wireless Sensor Networks. *IEEE Communications Surveys Tutorials*, 15:528–550, 2013.

[29] H. Erdem and V. Gungor. On the Lifetime Analysis of Energy Harvesting Sensor Nodes in Smart Grid Environments. *Elsevier Journal of Ad Hoc Networks*, 75-76:98 – 105, 2018.

[30] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient Network Flooding and Time Synchronization with Glossy. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2011.

[31] B. Firner, C. Xu, R. Howard, and Y. Zhang. Multiple Receiver Strategies for Minimizing Packet Loss in Dense Sensor Networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2010.

[32] L. Galluccio, G. Morabito, and S. Palazzo. TC-Aloha: A Novel Access Scheme for Wireless Networks with Transmit-only Nodes. *IEEE Transactions on Wireless Communications*, 12:3696–3709, 2013.

[33] M. H. S. Gilani, I. Sarrafi, and M. Abbaspour. An Adaptive CSMA/TDMA Hybrid MAC for Energy and Throughput Improvement of Wireless Sensor Networks. *Elsevier Journal of Ad Hoc Networks*, 11:1297 – 1304, 2013.

[34] D. D. Guglielmo, F. Restuccia, G. Anastasi, M. Conti, and S. K. Das. Accurate and Efficient Modeling of 802.15.4 Unslotted CSMA/CA through Event Chains Computation. *IEEE Transactions on Mobile Computing*, 15:2954–2968, 2016.

[35] L. Guntupalli, J. Martinez-Bauset, and F. Y. Li. Performance of Frame Transmissions and Event-Triggered Sleeping in Duty-Cycled WSNs with Error-Prone Wireless Links. *Elsevier Journal of Computer Networks*, 134:215 – 227, 2018.

[36] Z. Hadzi-Velkov and B. Spasenovski. Capture Effect in Wireless LANs with RAKE Reception of DSSS/DPSK Signals. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2004.

[37] K. Han and K. Huang. Wirelessly Powered Backscatter Communication Networks: Modeling, Coverage, and Capacity. *IEEE Transactions on Wireless Communications*, 16:2548–2561, 2017.

[38] P. Hu, P. Zhang, and D. Ganesan. Laissez-Faire: Fully Asymmetric Backscatter Communication. In *Proceedings of the ACM Conference on Special Interest Group on Data Communication (SIGCOMM)*, 2015.

[39] C. Huebner, S. Hanelt, T. Wagenknecht, R. Cardell-Oliver, and A. Monsalve. Long Range Wireless Sensor Networks Using Transmit-only Nodes. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2010.

[40] IHS Markit. The Internet of Things: a Movement, not a Market. URL: https://cdn.ihs.com/www/pdf/IoT_ebook.pdf, 2017.

[41] T. Istomin, A. L. Murphy, G. P. Picco, and U. Raza. Data Prediction + Synchronous Transmissions = Ultra-low Power Wireless Sensor Networks. In *Proceedings of the ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2016.

[42] S. Ji and Z. Cai. Distributed Data Collection in Large-Scale Asynchronous Wireless Sensor Networks Under the Generalized Physical Interference Model. *IEEE/ACM Transactions on Networking*, 21:1270–1283, 2013.

[43] X. Ji, Y. He, J. Wang, W. Dong, X. Wu, and Y. Liu. Walking down the STAIRS: Efficient Collision Resolution for Wireless Sensor Networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2014.

[44] A. C. Jose and R. Malekian. Improving Smart Home Security: Integrating Logical Sensing Into Smart Home. *IEEE Sensors Journal*, 17:4269–4286, 2017.

[45] G. Kaddoum, E. Soujeri, and Y. Nijsure. Design of a Short Reference Noncoherent Chaos-Based Communication Systems. *IEEE Transactions on Communications*, 64:680–689, 2016.

[46] H. Keong, K. Thotahewa, and M. Yuce. Transmit-only Ultra Wide Band Body Sensors and Collision Analysis. *IEEE Sensors Journal*, 13:1949–1958, 2013.

[47] R. Kuntz, J. Montavont, and T. Noël. Improving the Medium Access in Highly Mobile Wireless Sensor Networks. *Springer Journal on Telecommunication Systems*, 52:2437–2458, 2013.

[48] K. Langendoen and A. Meier. Analyzing MAC Protocols for Low Data-Rate Applications. *ACM Transactions on Sensor Networks*, 7:19, 2010.

[49] M. T. Lazarescu. Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 3:45–54, 2013.

[50] H. Lin, X. Liu, X. Wang, and Y. Liu. A Fuzzy Inference and Big Data Analysis Algorithm for the Prediction of Forest Fire Based on Rechargeable Wireless Sensor Networks. *Elsevier Journal of Sustainable Computing*, 18:101 – 111, 2018.

[51] Q. Lin, S. Mohan, and M. A. Weitnauer. Interference-insensitive Synchronization Scheme of MSK for Transmit-only Wireless Sensor Network. In *Proceedings of the IEEE International Conference on Communications (ICC)*, 2016.

[52] C.-J. Liu, P. Huang, and L. Xiao. TAS-MAC: A Traffic-Adaptive Synchronous MAC Protocol for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 12:1:1–1:30, 2016.

[53] X. Liu, Y. Zhu, L. Kong, C. Liu, Y. Gu, A. V. Vasilakos, and M. Wu. CDC: Compressive Data Collection for Wireless Sensor Networks. *IEEE Transactions on Parallel and Distributed Systems*, 26:2188–2197, 2015.

[54] S. Lucero. IoT Platforms: Enabling the Internet of Things. Whitepaper, IHS Markit, 2016.

[55] D. Markert, P. Parsch, and A. Masrur. Using Probabilistic Estimates to Guarantee Reliability in Crossroad VANETs. In *Proceedings of the ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet)*, 2017.

[56] Maxim Integrated. MAX2837 Datasheet. URL: https://datasheets.maximintegrated.com/en/ds/MAX2837.pdf.

[57] S. Moulik, S. Misra, and D. Das. AT-MAC: Adaptive MAC-Frame Payload Tuning for Reliable Communication in Wireless Body Area Networks. *IEEE Transactions on Mobile Computing*, 16:1516–1529, 2017.

[58] E. Natalizio, D. Cavalcanti, K. Chowdhury, and M. E. Said. Advances in Wireless Communication and Networking for Cooperating Autonomous Systems. *Elsevier Journal of Ad Hoc Networks*, 68:3–5, 2018.

[59] E. D. N. Ndih and S. Cherkaoui. Adaptive 802.15.4 Backoff Procedure to Survive Coexistence with 802.11 in Extreme Conditions. In *Proceedings of the IEEE Consumer Communications & Networking Conference (CCNC)*, 2016.

[60] T. M. C. Nguyen, D. B. Hoang, and T. D. Dang. A Software-Defined Model for IoT Clusters: Enabling Applications on Demand. In *Proceedings of the IEEE International Conference on Information Networking (ICOIN)*, 2018.

[61] F. Österlind, L. Mottola, T. Voigt, N. Tsiftes, and A. Dunkels. Strawman: Resolving Collisions in Bursty Low-Power Wireless Networks. In *Proceedings of the ACM International Conference on Information Processing in Sensor Networks (IPSN)*, 2012.

[62] D. Pal, S. Funilkul, N. Charoenkitkarn, and P. Kanthamanon. Internet-of-Things and Smart Homes for Elderly Healthcare: An End User Perspective. *IEEE Access*, 6:10483–10496, 2018.

[63] P. Parsch and A. Masrur. A Reliability-Aware Medium Access Control for Unidirectional Time-Constrained WSNs. In *Proceedings of the International Conference on Real Time and Networks Systems (RTNS)*, 2015.

[64] P. Parsch and A. Masrur. A Reliable MAC for Energy-Efficient WSNs in the Era of IoT. In *Proceedings of the Euromicro Conference on Digital System Design (DSD)*, 2016.

[65] P. Parsch and A. Masrur. A Slot Sharing TDMA Scheme for Reliable and Efficient Collision Resolution in WSNs. In *Proceedings of the ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2016.

[66] P. Parsch and A. Masrur. A Reliable MAC for Delay-Bounded and Energy-Efficient WSNs. In *Proceedings of the 23rd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2017.

[67] P. Parsch and A. Masrur. Accounting for reliability in Unacknowledged time-constrained WSNs. *ACM Transactions on Cyber-Physical Systems (TCPS)*, 101:96 – 110, 2018. under review (major revision).

[68] P. Parsch and A. Masrur. On Reliable Communication in Transmit-Only Networks for Home Automation. *Elsevier Journal of Network and Computer Applications (JNCA)*, 101:96 – 110, 2018.

[69] P. Parsch, A. Masrur, and W. Hardt. Designing Reliable Home-Automation Networks based on Unidirectional Nodes. In *Proceedings of the IEEE International Symposium on Industrial Embedded Systems (SIES)*, 2014.

[70] N. Pereira, B. Andersson, and E. Tovar. Widom: A Dominance Protocol for Wireless Medium Access. *IEEE Transactions on Industrial Informatics*, 3:120–130, 2007.

[71] B. Radunovic, H. L. Truong, and M. Weisenhorn. Receiver Architectures for UWB-Based Transmit-Only Sensor Networks. In *Proceedings of the IEEE International Conference on Ultra-Wideband (ICU)*, pages 379–384. IEEE, 2005.

[72] T. Rappaport. *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall PTR, 2001.

[73] W. Rhee. *Wireless Transceiver Circuits: System Perspectives and Design Aspects*. CRC Press, 2015.

[74] K. S., K. Ovsthus, and L. M. Kristensen. An Industrial Perspective on Wireless Sensor Networks x2014; A Survey of Requirements, Protocols, and Challenges. *IEEE Communications Surveys Tutorials*, 16:1391–1412, 2014.

[75] L. Sanchez, L. Munoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, and D. Pfisterer. SmartSantander: IoT experimentation over a smart city testbed. *Elsevier Journal of Computer Networks*, 61:217 – 238, 2014.

[76] G. Santagati, T. Melodia, L. Galluccio, and S. Palazzo. Medium Access Control and Rate Adaptation for Ultrasonic Intrabody Sensor Networks. *IEEE/ACM Transactions on Networking*, 23:1121–1134, 2015.

[77] M. Sellars and D. Kostas. Comparison of QPSK/QAM OFDM and Spread Spectrum for the 2-11 GHz PMP BWAS. Technical report, Adaptive Broadband Corp., 2000.

[78] B. Shrestha, E. Hossain, and K. W. Choi. Distributed and Centralized Hybrid CSMA/CA-TDMA Schemes for Single-Hop Wireless Networks. *IEEE Transactions on Wireless Communications*, 13:4050–4065, 2014.

[79] B. L. R. Stojkoska and K. V. Trivodaliev. A review of Internet of Things for smart home: Challenges and solutions. *Elsevier Journal of Cleaner Production*, 140:1454 – 1464, 2017.

[80] P. Suriyachai, U. Roedig, and A. Scott. A Survey of MAC Protocols for Mission-Critical Applications in Wireless Sensor Networks. *IEEE Communications Surveys Tutorials*, 14:240–264, 2012.

[81] R. Tavakoli, M. Nabi, T. Basten, and K. Goossens. Dependable Interference-Aware Time-Slotted Channel Hopping for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 14:3:1–3:35, 2018.

[82] Texas Instruments. CC2420 Datasheet. URL: http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[83] Texas Instruments. CC2545 Datasheet. URL: http://www.ti.com/lit/ds/symlink/cc2545.pdf.

[84] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[85] A. Varga. The OMNeT++ Discrete Event Simulation System. In *Proceedings of the European Simulation Multiconference (ESM)*, 2001.

[86] G. Wang, J. Yu, D. Yu, H. Yu, L. Feng, and P. liu. DS-MAC: An Energy Efficient Demand Sleep MAC Protocol with Low Latency for Wireless Sensor Networks. *Elsevier Journal of Network and Computer Applications*, 58:155 – 164, 2015.

[87] M. Weisenhorn and W. Hirt. Uncoordinated Rate-Division Multiple-Access Scheme for Pulsed UWB Signals. *IEEE Transactions on Vehicular Technology*, 54:1646–1662, 2005.

[88] M. Won, T. Park, and S. H. Son. Asym-MAC: A MAC Protocol for Low-Power Duty-Cycled Wireless Sensor Networks with Asymmetric Links. *IEEE Communications Letters*, 18:809–812, 2014.

[89] L. Xu, D. T. Delaney, G. M. P. OHare, and R. Collier. The Impact of Transmission Power Control in Wireless Sensor Networks. In *Proceedings of the IEEE International Symposium on Network Computing and Applications (NCA)*, 2013.

[90] X. Xu, R. Ansari, A. Khokhar, and A. V. Vasilakos. Hierarchical Data Aggregation Using Compressive Sensing (HDACS) in WSNs. *ACM Transactions on Sensor Networks*, 11:45:1–45:25, 2015.

[91] C. Yi, L. Wang, and Y. Li. Energy Efficient Transmission Approach for WBAN Based on Threshold Distance. *IEEE Sensors Journal*, 15:5133–5141, 2015.

[92] P. Zhang, J. Wang, K. Guo, F. Wu, and G. Min. Multi-Functional Secure Data Aggregation Schemes for WSNs. *Elsevier Journal of Ad Hoc Networks*, 69:86 – 99, 2018.

[93] Y. Zhang, B. Firner, R. Howard, R. Martin, N. Mandayam, J. Fukuyama, and C. Xu. Transmit Only: An Ultra Low Overhead MAC Protocol for Dense Wireless Systems. In *Proceedings of the IEEE International Conference on Smart Computing (SMARTCOMP)*, 2017.

[94] Y. Zhang, Y. Zhou, L. Gao, Y. Qian, J. Li, and F. Shu. A Blind Adaptive Tuning Algorithm for Reliable and Energy-Efficient Communication in IEEE 802.15.4 Networks. *IEEE Transactions on Vehicular Technology*, 66:8605–8609, 2017.

[95] J. Zhao, C. Qiao, R. S. Sudhaakar, and S. Yoon. Improve Efficiency and Reliability in Single-Hop WSNs with Transmit-Only Nodes. *IEEE Transactions on Parallel and Distributed Systems*, 24:520–534, 2013.

[96] W. Zhao and X. Tang. Scheduling Sensor Data Collection with Dynamic Traffic Patterns. *IEEE Transactions on Parallel and Distributed Systems*, 24:789–802, 2013.

[97] Y. Z. Zhao, C. Miao, M. Ma, J. B. Zhang, and C. Leung. A Survey and Projection on Medium Access Control Protocols for Wireless Sensor Networks. *ACM Journal of Computing Surveys*, 45:7:1–7:37, 2012.

[98] X. Zheng, Z. Cao, J. Wang, Y. He, and Y. Liu. Interference Resilient Duty Cycling for Sensor Networks Under Co-Existing Environments. *IEEE Transactions on Communications*, 65:2971–2984, 2017.

[99] S. Zhuo, Z. Wang, Y. Q. Song, Z. Wang, and L. Almeida. A Traffic Adaptive Multi-Channel MAC Protocol with Dynamic Slot Allocation for WSNs. *IEEE Transactions on Mobile Computing*, 15:1600–1613, 2016.

Appendix A. The DEEP protocol

A.1. Proof of Theorem 1

Let us assume that a node starts sending its sequence of n packets at time t_0 with a constant inter-packet time t_i. According to Lemma 1, the first $n-1$ packets can be intercepted by packets of the remaining nodes, if all nodes send their corresponding n packets within d_{max} and are only activated once in a time interval $t_{max} = 2 \cdot d_{max}$. In order that at least one packet of an i-th node safely reaches its receiver, it should be guaranteed that no other packet from any other node affects the n-th packet of the i-th node.

If the j-th node affects the first of packets from the i-th node, none of the subsequent packets from node j should affect the n-th packet of node i. That is, none of node j's packets should have any overlapping with the n-th packet of node i. Clearly, the n-th packet of node i is sent at time $t_0+(n-1)\cdot t_i$. Let us now assume that a packet of node j is also sent at time t_0 interfering with the first packet of node i. The remainder of $\frac{(n-1)\cdot t_i}{t_j}$ should allow for enough *space* to send a node j's packet before the n-th packet of node i start being sent, i.e., before $t_0+(n-1)\cdot t_i$: $\mod\left(\frac{(n-1)\cdot t_i}{t_j}\right) \geq l_{max}$. However, a packet of node j can still affect the first packet of node i, if this is sent at time $t_0 - l_{max} + \varepsilon$ or $t_0 + l_{max} - \varepsilon$ where ε is an infinitesimally small amount of time. This is because, in the latter case, there will be some amount of packet overlapping (given by ε) between node i and j. As a result of this, the remainder of $\frac{(n-1)\cdot t_i}{t_j}$ should allow for enough space to send a node j's packet with whatever initial overlapping between the first packet of node i and a packet of node j: $\mod\left(\frac{(n-1)\cdot t_i}{t_j}\right) \geq 2 \cdot l_{max}$. In a similar manner, if a packet of node j affects the second packet of node i that is sent at time $t_0 + t_i$, it should be guaranteed that none of the subsequent packets of node j can affect node i's n-th packet: $\mod\left(\frac{(n-2)\cdot t_i}{t_j}\right) \geq 2 \cdot l_{max}$. For any two nodes i and j where $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$, this translates in that $\mod\left(\frac{\alpha_i \cdot t_i}{t_j}\right) \geq 2 \cdot l_{max}$ has to hold for $1 \leq \alpha_i \leq n-1$. The theorem follows.

A.2. Proof of Theorem 4

According to Lemma 5, in order that at least one packet of a node i reaches its destination in the worst case, it needs to send $k_i = n$ packets under the delayed-activation scheme. Taking a node j into consideration with $i \neq j$, Lemma 5 assumes that suitable inter-packet times t_i and t_j can be found such that node j cannot interfere with more than one packet within any node i's sequence of packets.

Now, for t_i and t_j to comply with Lemma 5, if the first packet of a node j's sequence intercepts a packet of node i, none of the subsequent node j's packets should affect another packet of the same node i's sequence, independent of which node i's packet has been first interfered by node j.

Let us assume that node i starts sending its sequence of n packets at time t_0 with a constant inter-packet time t_i. Clearly, the n-th packet of node i is sent at time $t_0+(n-1)\cdot t_i$. Let us further assume that a packet of node j is also sent at time t_0 interfering with the first packet of node i. In addition, let us assume that node j continuously sends packets in $[t_0, t_0+(n-1)\cdot t_i]$ following our delayed-activation scheme, i.e., node j's packets are separated by an integer multiple of t_j.

In order that the n-th packet of node i is not affected by node j, the remainder of $\frac{(n-1)\cdot t_i}{t_j}$ must allow for enough *space* to send a node j's packet before the n-th packet of node i starts being sent, i.e., before $t_0 + (n-1)\cdot t_i$: $\bmod\left(\frac{(n-1)\cdot t_i}{t_j}\right) \geq l_j$.

On the other hand, a packet of node j can still affect the first packet of node i, if this is sent at time $t_0 - l_j + \varepsilon$ or $t_0 + l_i - \varepsilon$ where ε is an infinitesimally small amount of time. This is because there will be some amount of packet overlapping (given by ε) between the corresponding packets of node i and j. As a result, the remainder of $\frac{(n-1)\cdot t_i}{t_j}$ should allow for enough space to send a node j's packet considering all possible initial overlapping between the first packet of node i and the packet of node j: $\bmod\left(\frac{(n-1)\cdot t_i}{t_j}\right) \geq l_i + l_j$.

In a similar manner, for the $(n-1)$-th packet of node i not to be affected by node j, the following has to hold: $\bmod\left(\frac{(n-2)\cdot t_i}{t_j}\right) \geq l_i + l_j$. For any two nodes i and j where $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$, this translates in that $\bmod\left(\frac{\alpha_i\cdot t_i}{t_j}\right) \geq l_i + l_j$ has to hold for $1 \leq \alpha_i \leq n-1$. The theorem follows.

Note that, if $\left\lfloor\frac{(n-1)\cdot t_i}{t_j}\right\rfloor = 0$ holds, node i can only be interfered once by node j, which is already considered in Lemma 5. As a result, if $\left\lfloor\frac{(n-1)\cdot t_i}{t_j}\right\rfloor = 0$ holds for all i and j, Lemma 5 becomes necessary and sufficient for a reliable communication.

A.3. Deriving $q_i(x)$

This section covers the derivation of $q_i(x)$, i.e., the probability that exactly x out of k_i packets of node i are lost due to internal interference. To this end, we consider different exemplary combinations of n, k, i and x to stepwise show the assumptions made to derive (4.3.1) and (4.3.2).

Example 1. Let us first consider the simple case of each node sending just a single packet per sequence $k_i = 1\ \forall i$. Further, we set $n = 4$, $i = 4$ and $x = 1$. The probability of losing this packet can be calculated as:

$$\begin{aligned} q_4(1) = {} & [\sigma_1\bar{\sigma}_2\bar{\sigma}_3 + \sigma_2\bar{\sigma}_1\bar{\sigma}_3 + \sigma_3\bar{\sigma}_1\bar{\sigma}_2] \\ & + [\sigma_1\sigma_2\bar{\sigma}_3 + \sigma_1\sigma_3\bar{\sigma}_2 + \sigma_2\sigma_3\bar{\sigma}_1] \\ & + [\sigma_1\sigma_2\sigma_3], \end{aligned} \tag{A.3.1}$$

where $\bar{\sigma}_j = 1 - \sigma_j$ is the probability of having no interference by node j with $1 \leq j \leq 3$.

In more detail, (A.3.1) is composed of 3 different parts, separated by square brackets. The first part contains the probabilities that node 4 has only 1 collision with any other node, for example, $\sigma_1\bar{\sigma}_2\bar{\sigma}_3$ means that there is a collision with node 1, but not with node 2 and 3. The second part considers double collisions, i.e., when interference is caused by 2 nodes simultaneously. And finally, the third part contains triple collisions. Note that node 4 must be transmitting for possible interference, hence, $q_4(x)$ does not depend on σ_4.

Calculating $q_i(x)$ is complex, since all combinations of collisions must be regarded. This complexity further increases for higher n, k and x, hence, in order to simplify calculations, let us consider:

$$q_4(1) \leq \sigma_1 + \sigma_2 + \sigma_3. \tag{A.3.2}$$

This equation only sums up probabilities σ_j of having a (single) collision with any node j. Multiple, simultaneous collisions, on the other hand, are included in the form of intersections

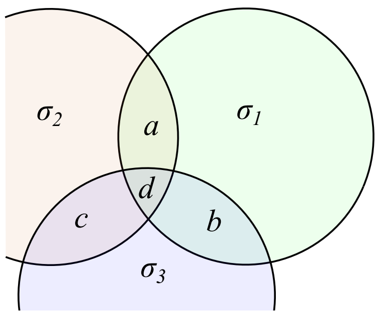


Figure A.1.: A Venn diagram showing the relation between collision probabilities of three different nodes. The different sets represent the probabilities of having interference with one node, e.g., σ_1 means that there is a collision with node 1. The intersections a, b and c consider double collisions, i.e., an interference with two nodes simultaneously. Finally, d accounts for the case of a triple collision.

between any of those σ_j — see a, b, c, d in Fig. A.1. However, by simply adding all σ_j, some of these intersections are included multiple times. In case of (A.3.2), a, b and c are added twice and d three times, therefore, we calculate a $q_i(x)$ that is always greater than the actual (correct) value. In general, these intersections are very small, since $\sigma_j << 1$ holds, and, hence, this error does not significantly affect results.

Clearly, for (A.3.2) to be valid, the following must hold:

$$\sum_{\forall i} \sigma_i \leq 1. \tag{A.3.3}$$

This is a logical assumption, since the *utilization* of the communication channel cannot exceed 100 %, otherwise reliable communication is not possible anymore. Note that periods found by Alg. 1 comply with (A.3.3).

Example 2. Next, we consider the case of two packets being sent per sequence, i.e., $k_i = 2\ \forall i$, $n = 4$ and $x = 1$. This yields the following expressions:

$$\begin{aligned} q_4(1) &\leq [\sigma_1 + \sigma_2 + \sigma_3] + [\bar{\sigma}_1\sigma_1 + \bar{\sigma}_2\sigma_2 + \bar{\sigma}_3\sigma_3], \\ &\leq \sigma_1(1 + \bar{\sigma}_1) + \sigma_2(1 + \bar{\sigma}_2) + \sigma_3(1 + \bar{\sigma}_3). \end{aligned} \tag{A.3.4}$$

The terms in the square brackets in (A.3.4) describe the collision probabilities for each packet within node 4's sequence. For example, σ_1 in the first term describes the probability that node 4's first packet collides with a packet of node 1. If that happens, the probability of the second packet interfering again with node 1 is zero ($\bar{\sigma}_1 = 1$). In case of the second term, however, we have to account for all previously sent packets: For example, $\bar{\sigma}_1\sigma_1$ in the second term describes the chance that node 1 does not interfere with the first, but with the second packet.

Again, we can simplify the calculation of $q_i(x)$:

$$q_4(1) \leq 2\sigma_1 + 2\sigma_2 + 2\sigma_3. \tag{A.3.5}$$

Since we know the ratios of packet lengths to period times are very small, i.e., $\sigma_i \ll 1$, we can approximate the probability of missing a packet $\bar{\sigma}_i = (1 - \sigma_i) \approx 1$. By doing so, we calculate a greater and more pessimistic $q_i(x)$, however, the error is again very small. In order for (A.3.5) to be valid, the following must hold:

$$\sum_{\forall i} k_i\sigma_i \leq 1. \tag{A.3.6}$$

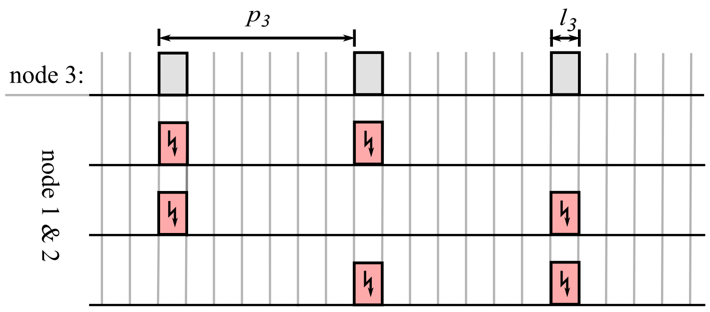


Figure A.2.: Possible combinations of interference that lead to 2 packet collisions within a sequence of node 3: Either packets 1 and 2, packets 1 and 3 or packets 2 and 3 collide with packets of node 1 and 2. Further, the order in which nodes collide, e.g., node 1 first and node 2 second or vice versa, must be regarded as well, leading to 6 possible combinations in total.

This is a necessary condition in order that the simplification by (A.3.5) does not result in a probability that is greater than one. Since periods by Alg. 1 increase with k and n, (A.3.6) usually holds making the use of the simplified (A.3.5) possible.

Example 3. Let us consider the effects of changing the number of packets x that are allowed to be lost. To this end, we set our network parameters to $x = 2$, $n = 3$, $i = 3$ and $k_i = 3\ \forall i$.

Taking the simplifications of (A.3.2) and (A.3.5) into account, the probability of losing 2 packets within a sequence can be upper bounded by $\sigma_i \sigma_j$ with $j \neq i$. In other words, this is the probability of losing a first *and* a second packet. As illustrated in Fig. A.2, there are 6 possible combinations of losing 2 out of 3 packets, which can be computed using the binomial coefficient. We have $\binom{3}{2} = 3$ possibilities of being first interfered by node 1 and then by node 2 and another $\binom{3}{2} = 3$ in the reverse case. This results in:

$$q_3(2) \leq \binom{3}{2} (\sigma_1\sigma_2 + \sigma_2\sigma_1) = 6\,\sigma_1\sigma_2. \tag{A.3.7}$$

Let us now generalize $q_i(x)$ for an arbitrary number of packets sent k_i, packets lost x and number of nodes n. This results in (4.3.1) as stated in Section 4.3:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j\neq i}^{n} \sigma_j \left(\sum_{l=1;\ l\neq i,j}^{n} \sigma_l \cdots \right) \right), \tag{A.3.8}$$

where $1 \leq x \leq (n-1)$ and $q_i(n) = 0\ \forall i$. The number of summations equals x. Again, the first part is the binomial coefficient and accounts for the different combinations of packets that collide within the sequence. The remaining part considers the combinations of different nodes that can cause collisions within the sequence. By assuming $\sigma_i = \sigma\ \forall i$, (A.3.8) can be further simplified to:

$$\begin{aligned} q_i(x) &\leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j\neq i}^{n} \sigma \left(\sum_{l=1;\ l\neq i,j}^{n} \sigma \cdots \right) \right), \\ &\leq \binom{k_i}{x} \cdot \Big((n-1)\sigma \Big((n-2)\sigma \cdots \Big) \Big), \\ &\leq \binom{k_i}{x} \cdot \left(\prod_{j=1}^{x} (n-j) \right) \cdot \sigma^x. \end{aligned} \tag{A.3.9}$$

This equation allows to calculate the transmission reliability for each node i , if it sends a reduced number of $k_i \leq n$ packets. This allows us to adjust the packet numbers individually for each node to save energy whenever data loss can be tolerated by the application.

Appendix B.

Extending DEEP and RARE to bidirectional communication

In the course of this work it was found that the principles of DEEP and RARE can also be effectively used in bidirectional networks. This section introduces two enhanced versions of DEEP and RARE, called bi-DEEP and bi-RARE that extend their functionality by implementing carrier sensing and acknowledgments:

- **bi-DEEP** (bidirectional deterministic protocol) guarantees full reliability by sending data as sequences of redundant packets with constant inter-packet times. In contrast to DEEP, carrier sensing is used to reduce collisions and shorten delays, and ACKs to decrease the number of transmitted packets on average.
- **bi-RARE** (bidirectional random reliable protocol) can be configured to a user-specified reliability similar to RARE, for which it transmits sequences of redundant packets in random time intervals. However, it uses carrier sensing to reduce collisions and ACKs to decrease the average number of packets transmissions.

By using carrier sensing, these two protocols can efficiently avoid collisions. For example, if a node detects an ongoing transmission, it will not try to send — this would most likely destroy both packets — but will instead wait for another inter-packet time before trying again. Further, acknowledgments reduce the average number of packets transmitted per node, since all pending (redundant) data packets can be skipped after an ACK is received. As a result, these two mechanisms reduce collisions and traffic load, which results in a better energy consumption and reliability. However, compared to the unidirectional protocols DEEP and RARE, complexity and overhead is higher, which might cause problems in some scenarios. In particular during high network traffic, this can possibly lead to increased collision numbers, as analyzed in the case of RARE and bi-RARE in Section 6.4.6.

Note that the focus of this work is on the unidirectional protocols DEEP and RARE. The bi-DEEP and bi-RARE extensions shown in this appendix are only included for the sake of completeness, in order to give a prospect of possible (bidirectional) extensions. On this basis, these two protocols are derived only as simplified models, i.e., without consideration of more complex factors such as arbitrary node types or clock drift and external interference that would otherwise go beyond the scope of this work. Note that both protocols have been published as conference papers, i.e., bi-DEEP in [64] and bi-RARE in [66].

Regarding their models and assumptions, both bi-DEEP and bi-RARE are based on the same setting as listed in Section 1.3. That is, the network consists of n transmit-only nodes and multiple receive-only sinks that are connected in a single-hop (star-topology) fashion. Data transfers can either be triggered by events or occur periodically, and at least one packet must arrive within a deadline d_{max} at the sink to ensure normal (error-free) operation. Lastly, it is assumed that there is no external interference and nodes can always detect each others transmissions, i.e., there are no hidden terminals (see Section 2.2.3).

Now, before starting with the working principles of bi-DEEP and bi-RARE, the following section first examines the optimal carrier sensing duration and discusses the timings of packet transmissions in more details.

B.1. Carrier sensing duration and packet timings

As discussed before in Section 2.4, communication in bidirectional networks with carrier sensing and ACK differs significantly from communication in unidirectional networks. This section further discusses these differences, in particular, the different packet timings, and derives the optimal carrier sense duration.

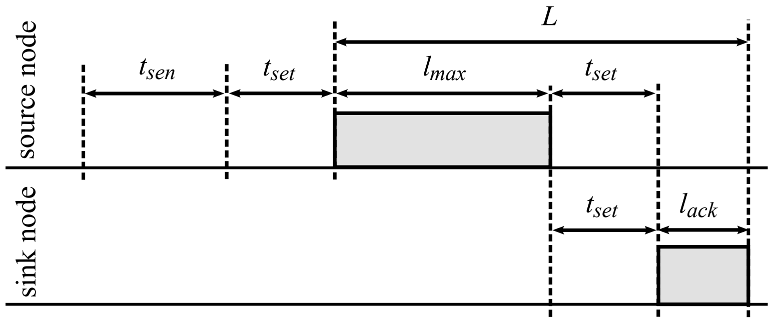


Figure B.1.: Timing of a successful packet transmission.

Let us now have a look at the timings of a successful packet transmission with carrier sensing and ACK, as shown in Fig. B.1. In contrast to a unidirectional network, where a node starts transmitting directly, in a bidirectional setting, a node always senses the channel for t_{sen} time first before trying to transmit. In case the channel is blocked, i.e., another node is currently transmitting, it will not try to transmit, but instead back off and wait for another inter-packet time before trying again. In case the channel is free, the node switches its transceiver from receive to transmit mode in t_{set} time to be able to send its data packet. After transmission, the node switches back to receive mode to be able to receive an ACK from the sink. Similarly, the sink also performs a mode switch to transmit the ACK, which again takes t_{set} time. Note that t_{set} includes possible latencies such as processing and propagation delays.

Once the acknowledgment has been sent, the sink node needs t_{set} additional time to switch back to receive mode and be able to receive further packets. However, since there cannot be any packet in this last t_{set}, it can be safely neglected. That is, another node can start transmitting at earliest $t_{sen}+t_{set}$ time after the channel is free, i.e., after the acknowledgment has been sent — otherwise it would detect the ACK and back off. As a consequence, L denotes the total delay incurred from the start of a transmission by the source node to the time at which the sink node finishes its acknowledgment — see Fig. B.1:

$$L = l_{max} + l_{ack} + t_{set}. \tag{B.1.1}$$

Next, let us analyze the optimal length of the sensing period t_{sen}. To this end, it has to be considered that every transceiver IC has a specific sensitivity, i.e., it requires a certain amount of time to detect whether a channel is busy or not. This is typically the time to receive a few bits at a given transmission speed, which is denoted by $\bar{t}$ in the following. Note that a signal must be present for at least $\bar{t}$ *continuously* on the channel for a node to detect it reliably.

The shortest possible sensing interval can be as short as $t_{sen} = \bar{t}$. However, as displayed in Fig. B.1, there is a gap of size t_{set} in between data packets and the corresponding acknowledgments. Now, in order to not falsely detect the channel as free during this time, t_{sen} should be chosen such that:

$$t_{sen} = t_{set} + 2\bar{t}. \tag{B.1.2}$$

By adding $2\bar{t}$ in (B.1.2), it is ensured that t_{sen} always overlaps by at least $\bar{t}$ *continuously* with either the data or acknowledgment packet. Note that a longer t_{sen} than (B.1.2) has no benefit, but it rather increases energy consumption of the node without guaranteeing better results.

Note that, although carrier sensing greatly reduces the chance of collisions, it cannot fully prevent them. That is, since nodes can be triggered by independent events, it may happen that the carrier sense intervals of two or more nodes overlap such that they cannot see each other. For example, let us assume that a node 1 and a node 2, sense the communication channel for $50\,\mu$s and $10\,\mu$s respectively, before they start transmitting. It can happen that node 2 is triggered $40\,\mu$s after node 1 such that their sensing intervals do not overlap with the other's packet transmission. As a result, they do not detect each other and packets still get lost.

In summary, neither ACKs nor carrier sensing can fully prevent collisions and thus do not enable reliable communication per se. For this reason, a more elaborate medium access control is required as presented by the bi-DEEP and bi-RARE protocols in the following.

B.2. The bi-DEEP protocol

Similar to the unidirectional protocol DEEP, bi-DEEP enables fully reliable communication between nodes in a network, i.e., it can guarantee that data always reaches its destination in the worst case. However, in order to reduce the high costs of full reliability, i.e., the increased packet numbers and prolonged inter-packet times, bi-DEEP implements carrier sensing and acknowledging mechanisms. That is, ACKs can decrease packet numbers, since they indicate a successful reception, after which the node can cancel all pending (redundant) retransmissions, and carrier sensing reduces collision numbers and helps shortening inter-packet times.

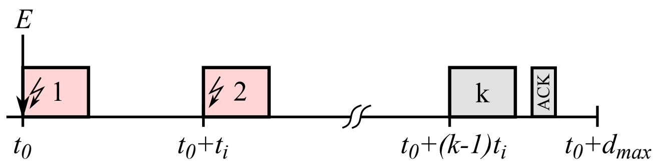


Figure B.2.: Working principle of bi-DEEP: After being activated by an event E, node i starts transmitting redundant data packets with constant inter-packet times t_i until an ACK is received — here for the k-th packet. In the worst case, up to k packets are sent within a deadline d_{max}.

Fig. B.2 depicts the working principle of bi-DEEP in more details. Whenever a node i is triggered by an event E, it starts transmitting a sequence of redundant packets with constant inter-packet times t_i until an ACK is received or a maximum number of retransmissions k is reached. Here, $0 \leq k \leq n$, $0 \leq i \leq n$ and $t_i \in \mathbb{R}_{>0}$ hold. Transmitting a packet with ACK requires L time (see Fig. B.1) and at least one packet of a sequence must arrive within a deadline d_{max} from the time of being activated; both parameters are common to all nodes.[1] To reduce collision numbers, carrier sensing is performed before each transmission, i.e., if the channel is busy, the node backs off, otherwise it continues as shown in Fig. B.1.

The parameters n, d_{max}, l_{max} are determined by the application and known in advance. The packet numbers k, inter-packet times t_i and minimum activation time t_{act}, on the other hand, must still be selected. These are derived analytically in the following sections.

B.2.1. Selecting packet numbers

Let us first start with deriving the number of packets k that must be transmitted in the worst-case to guarantee reliable communication. Towards this, Theorem 5 establishes a relation between inter-packet times of two nodes.

[1]Again, this appendix only covers basic models of bi-DEEP and bi-RARE that assume there is only one type of node in the network and neglect practical factors such as external interference and clock drift.

Theorem 5. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $d_{max} \leq t_{act}$. For any node i in the network, it can be guaranteed that at most one packet is interfered on the communication channel by another node j, if the following condition holds for $1 \leq i \leq n$, $1 \leq j \leq n$, $1 \leq \alpha \leq k$, and $i \neq j$:*

$$\operatorname{mod}\left(\frac{\alpha \cdot t_i}{t_j}\right) \geq 2(t_{set} + \bar{t}), \tag{B.2.1}$$

where t_{set} has been selected as per (B.1.2) *and is the time to switch between send and receive mode at the nodes, $\bar{t}$ is the sensitivity of the node as defined above, while t_i and t_j are the (constant) inter-packet times of node i and j respectively.*

Proof. Let us assume that any node i with $1 \leq i \leq n$ starts sending its first packet at time t_0, i.e., it is triggered by $t_0 - t_{sen} - t_{set}$. If this packet of node i is interfered by a packet of j being sent at the same time, to prove this theorem, we need to guarantee that none of the other potential transmissions by node i can be interfered anew by node j. Recall that nodes are activated only once within t_{act}, and hence, if they send more than one packet, these are due to retransmissions.

As a result, the subsequent activation times of node j for packet transmission need to be such that either node i detects node j or vice versa when they sense the communication channel. Now, if node i starts sending a packet, recall that a node j will be able to detect it, if t_{sen} is selected as per (B.1.2). From Fig. B.1, note that node j will not be able to detect a node i's packet transmission, if node i starts sending less than $\bar{t}$ time before the end of node j's t_{sen} – recall that $\bar{t}$ is the least amount of time a signal needs to be present for a node to detect it. From this point in time and until the end of the following t_{set}, node j is *blind*, i.e., in an interval of length $\bar{t} + t_{set}$.

Similarly, node i is unable to detect a node j's packet transmission, if node j starts sending less than $\bar{t}$ time before the end of node i's t_{sen}. This again result in an interval of length $\bar{t} + t_{set}$ in which node i is *blind*. As a consequence, node i and j will interfere with each other at the communication channel, only if their activation times fall into an interval of length $2(\bar{t} + t_{set})$ from one another.

If node i and j interfere with each other at the communication channel, their subsequent packet transmissions can be prevented from interfering by properly selecting t_i and t_j. In particular, the activation times of node i and j need always to be separated by at least $2(\bar{t} + t_{set})$ time, which leads to (B.2.1) and the theorem follows. □

Theorem 5 ensures that any two nodes i and j interfere only once with each other, i.e., their packets collide at most once per activation. However, it does not state how often nodes can be activated within t_{act} and how this affects collision between subsequent activations. To this end, let us consider the following analysis.

Lemma 7. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $d_{max} \leq t_{act}$. If $t_{act} = d_{max}$ holds, i.e., a node can immediately start transmitting anew after it finished its sequence, at least $2(n-1)$ packets of any node i will be lost in the worst case, independent of the inter-packet separation t_i with which packets are transmitted.*

Proof. Let us assume that node j is activated at time t_0 and, hence, sends up to k packets – depending on whether packets need to be retransmitted or not – within $[t_0, t_0 + d_{max}]$ with a constant inter-packet separation t_j. Further, let us assume that node j's last packet is sent at time $t_0 + d_{max} - l_{max}$ such that this packet is fully transmitted by $t_0 + d_{max}$. If node i

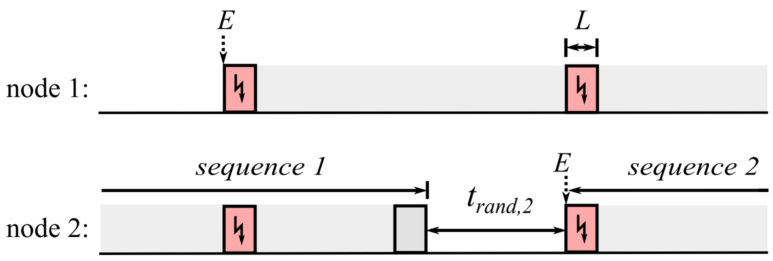


Figure B.3.: Illustration of Lemma 7 for the case of two nodes. Upon activation by an event E, node 1 starts transmitting its first packet, which is interfered by a packet of node 2. Since node 2 finishes its sequence before the second packet of node 1 is transmitted and there is no transmission pause after a sequence, i.e., $d_{max} = t_{act}$, it can happen that node 2 is triggered again at a time $t_{rand,2}$, such that there is a second collision with node 1.

starts transmitting at time $t_0 + d_{max} - l_{max}$, i.e., it was activated $t_{sen} + t_{set}$ time before — see Fig. B.1 — and did not detect a transmission by node j in t_{sen}, the last packet of node j and the first packet of node i will be lost.

In the worst case, the remaining $n - 2$ nodes in the network start transmitting at $t_0 + d_{max} - l_{max} + \alpha \cdot t_i$ where $1 \leq \alpha \leq k$ is an integer number and t_i is node i's inter-packet separation. As a consequence, $n-1$ packets of node i will be lost independent of inter-packet times of node i, of node j, and of the other $n - 2$ nodes.

Similarly, node j can interfere with further packets of node i, if it is activated anew before time $t_0 + 2d_{max} - l_{max}$, i.e., before node i finishes transmitting its k packets. Since we consider $t_{act} = d_{max}$, i.e., a node can immediately start transmitting anew after it finished its sequence, it is possible that, in the worst case, node j interferes with up to two packets from each other node, i.e., up to $2(n - 1)$ packets can be lost – see Fig. B.3. The lemma follows. □

As a result of Lemma 7, in the worst case, each node might need to transmit up to $2(n-1)+1 = 2n-1$ packets within d_{max} and should only be activated once in an interval of length t_{act} with $t_{act} = d_{max}$. To decrease this pessimism and reduce k, a transmission pause is introduced as stated in the following corollary:

Corollary 3. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $d_{max} \leq t_{act}$. If $t_{act} \geq 2d_{max}$ holds, i.e., a node will wait for an inter-sequence pause of at least d_{max} after each sequence before transmitting anew, at least $n - 1$ packets of any node i will be lost in the worst case, independent of the inter-packet separation t_i with which packets are transmitted.*

Proof. Immediate from Lemma 7. □

In summary, it can be observed that according to Lemma 7, the missing pause after a sequence leads to an additional $n - 1$ unavoidable packet losses per sequence compared to Corollary 3. This means that, on the one hand, the missing pause allows triggering the node more frequently and, hence, theoretically increases the total data throughput. On the other hand, the energy efficiency is decreased due to a higher number of possible packet collisions in the worst case. However, it was found out that the higher packet collision rate mostly dominates and finally leads to a lower data throughput that is comparable to a system with inter-sequence pauses. Since energy is a crucial factor for WSN, it is therefore assumed that the system implements a transmission pause after each sequence as stated in Corollary 3.

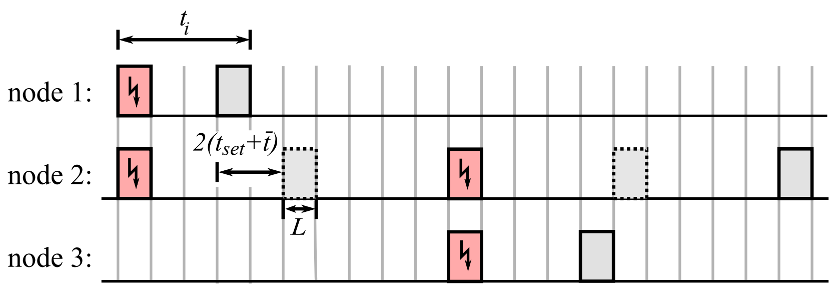


Figure B.4.: Illustration of Lemma 8 for the case of three nodes, i.e., $n = 3$. There can be $n - 1 = 2$ packet collisions on the communication. In addition, since (B.2.1) holds according to Theorem 5, nodes will be able to detect transmissions of one another. As a result, in the worst case, one node's packet transmission can be delayed additional $n - 1 = 2$ times (second node's dotted packets).

Corollary 3 states that there are at least $n - 1$ unavoidable packet losses per sequence in the worst case. However, in order to derive *safe* values for k, also the effects of carrier sensing need to be considered, i.e., when nodes skip packets due to a blocked channel. This is analyzed in the following lemma.

Lemma 8. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $t_{act} \geq 2d_{max}$. For any two nodes i and j in the network, t_i and t_j have been selected such that they comply with* (B.2.1)*. If interference from outside the network can be neglected, the following number of transmissions k suffices to guarantee reliability:*

$$k = 2n - 1. \tag{B.2.2}$$

Proof. Theorem 5 states that any two nodes i and j in the network only interfere once with each other, if t_i and t_j are selected as per (B.2.1). This means that, in the worst case, a packet of a node i can be interfered $n - 1$ times by the remaining $n - 1$ nodes. On the other hand, if t_i and t_j comply with (B.2.1), they will be able to detect transmissions of one another. However, in the worst case, a packet of a node i can also be delayed $n - 1$ times by transmissions of the remaining $n - 1$ nodes. The maximum number of retransmission attempts is hence $2(n - 1)$, i.e., in the worst case, a node may try to send a packet a total number $k = 2(n - 1) + 1 = 2n - 1$ times before being successful – Fig. B.4 illustrates this for $n = 3$. The lemma follows. □

B.2.2. Calculating inter-packet times

The following analysis derives *safe* values for t_i for any i and $1 \leq i \leq n$. Towards this, Lemma 9 guarantees that there is at most one collision between any two nodes.

Lemma 9. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $t_{act} \geq 2d_{max}$. In order to guarantee that at most one packet of a node i is interfered on the communication channel by another node $i - 1$, the following condition must hold for any t_i and t_{i-1} where $1 < i \leq n$:*

$$t_i - t_{i-1} \geq 2(t_{set} + \bar{t}), \tag{B.2.3}$$

given that $t_{i-1} < t_i < 2t_{i-1}$ holds, i.e., $\lfloor \frac{t_i}{t_{i-1}} \rfloor = 1$.

Proof. According to Theorem 5, if (B.2.1) holds for all i and j where $i \neq j$ and $1 \leq \alpha \leq k$, it can be guaranteed that any two nodes i and j interfere only once with each other.

Now, for a any i, $j = i-1$, and $\alpha = 1$, from (B.2.1) we have $\bmod\left(\frac{t_i}{t_{i-1}}\right) \geq 2(t_{set} + \bar{t})$, which again has to hold according to Theorem 5. Since $t_i > t_{i-1}$ and $\lfloor \frac{t_i}{t_{i-1}} \rfloor = 1$ hold for $1 < i \leq n$, we have that $t_i - t_{i-1} \geq 2(t_{set} + \bar{t})$ and the lemma follows. □

Lemma 10. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $t_{act} \geq 2d_{max}$. If one packet of a node i is interfered by a packet of node j, in order to guarantee that the next packet sent by node i is not interfered again by node j, the following condition must hold for the minimum inter-packet time $t_{min} = \min\limits_{1 \leq i \leq n}(t_i)$:*

$$t_{min} \geq L + t_{set} + t_{sen}. \tag{B.2.4}$$

Proof. Let us assume that node i is triggered at time t_0 and successfully sends its data to the corresponding sink. This means that the node is busy until $t_0 + t_{sen} + t_{set} + L$ – see Fig. B.1. If the next packet of node i starts directly afterwards, i.e., without any further waiting time, the waited period time is $t_i = L + t_{set} + t_{sen}$, as stated in (B.2.4).

Let us again assume node i is triggered at time t_0, but this time its first packet is interfered by a packet of node j. According to Theorem 5 that node j was triggered in a time interval $[t_0-(t_{set}+\bar{t}), t_0+(t_{set}+\bar{t})]$. This means that after the (collided) packet of node i is transmitted, node j can still occupy the channel for up to $t_{set}+\bar{t}$ time until $t_0+t_{sen}+t_{set}+l_{max}+(t_{set}+\bar{t})$ – see Fig. B.1. Both nodes will now listen for acknowledgments, which are not sent since packets were corrupted. This takes additional $2t_{set} + l_{ack}$ time, which is always greater than the maximum possible delay $(t_{set} + \bar{t})$ of node j. As a consequence, the earliest point in time when node i can start with its second packet is $t_0 + L + t_{set} + t_{sen}$, which results in a t_i equivalent to (B.2.4). Due to (B.2.3), there will be no further collision with node j. The lemma follows. □

Lemma 11. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $t_{act} \geq 2d_{max}$. The (constant) inter-packet time is upper bounded by $t_{max} = \max\limits_{1 \leq i \leq n}(t_i)$:*

$$t_{max} \leq \frac{d_{max} - (t_{sen} + t_{set} + L)}{k}. \tag{B.2.5}$$

Proof. Without loss of generality, let us assume that node i is activated at time t_0. In order that n packets can be sent within $[t_0, t_0 + d_{max}]$, the n-th packet has to start at latest at $t_0 + d_{max} - (t_{sen} + t_{set} + L)$. This way, the sink node finished switching back to receive mode after acknowledging node i's n-th packet exactly at $t_0 + d_{max}$. □

As the upper and lower bound of inter-packet times have been determined in the previous analysis, the following theorem now derives a formula to find *safe* values for t_i for any i and $1 \leq i \leq n$.

Theorem 6. *Let us consider a set of n independent transmit-only nodes, which are activated once within a time interval of length t_{act} and transmit a sequence of k packets within d_{max} where $t_{act} \geq 2d_{max}$. If the first packet of a node i is interfered by a packet of node j, in order to guarantee that the next $(2n-2)$ packets sent by node i are not interfered again by node j, the following condition must hold for the minimum inter-packet time $t_{min} = \min\limits_{1 \leq i \leq n}(t_i)$:*

$$t_{min} \geq (2n-2) \cdot (n-1) \cdot 2(t_{set} + \bar{t}) + 2(t_{set} + \bar{t}), \tag{B.2.6}$$

where as before $t_{min} < t_i < 2t_{min}$ holds, i.e., $\lfloor \frac{t_i}{t_{min}} \rfloor = 1$, for $1 \leq i \leq n$. In addition, $\lfloor \frac{k \cdot t_{min}}{(k-1) \cdot t_{max}} \rfloor = 1$ also holds, i.e., $(k-1) \cdot t_{min} < (k-1) \cdot t_{max} < k \cdot t_{min}$, for $1 < k \leq n-1$ and $t_{max} = \max\limits_{1 \leq i \leq n} (t_i)$.

Proof. Let us again assume that the first packet sent by a node i is interfered at time t_0 by a packet of node j. In order that the next $(2n-2)$ packets sent by node i are not interfered again by node j, (B.2.1) needs to hold for all i and j where $i \neq j$ and $1 \leq k_i \leq n-1$ as per Theorem 5.

Without loss of generality, let us assume that all t_i are sorted in order of increasing values, i.e., $t_i > t_j$ if $i > j$. Hence $t_{min} = \min\limits_{1 \leq i \leq n} (t_i) = t_1$ and $t_{max} = \max\limits_{1 \leq i \leq n} (t_i) = t_n$ hold.

Let us first consider $i = 1$ and $j = n$. If $\alpha = 1$ holds, from (B.2.1) we have that $\bmod\left(\frac{t_1}{t_n}\right) \geq 2(t_{set} + \bar{t})$ is equal to $t_1 \geq 2(t_{set} + \bar{t})$ since $t_1 < t_n$. For $\alpha = 2$, from (B.2.1) we have that $\bmod\left(\frac{2t_1}{t_n}\right) \geq (t_{set} + \bar{t})$ is equal to $2t_1 - t_n = t_1 - 2(n-1) \cdot (t_{set} + \bar{t})$, as $\lfloor \frac{2t_1}{t_n} \rfloor = 1$ holds – see again proof of Lemma 9. Similarly, for $\alpha = 3$, we have that $\bmod\left(\frac{3t_1}{t_n}\right) \geq (t_{set} + \bar{t})$ is equal to $3t_1 - 2t_n = t_1 - 2 \cdot 2(n-1) \cdot (t_{set} + \bar{t}) \geq 2(t_{set} + \bar{t})$, as $\lfloor \frac{3t_1}{2t_n} \rfloor = 1$ holds. For $\alpha = 2n-1$, we have that $\bmod\left(\frac{(2n-1) \cdot t_1}{t_n}\right) \geq 2(t_{set} + \bar{t})$ is equal to $(2n-1) \cdot t_1 - (2n-2) \cdot t_n = t_1 - (2n-2) \cdot 2(n-1) \cdot (t_{set} + \bar{t}) \geq 2(t_{set} + \bar{t})$, as $\lfloor \frac{(2n-1) \cdot t_1}{(2n-2) \cdot t_n} \rfloor = 1$ also holds. As a result, we have that $t_1 \geq (2n-2) \cdot (n-1) \cdot 2(t_{set} + \bar{t}) + 2(t_{set} + \bar{t})$ which is lower bound for $t_{min} = t_1$ stated in (B.2.5). Since $t_n = t_{max}$, note that choosing another j where $1 < j < n$ yields a lower bound that is closer to that of Lemma 9. In other words, the lower bound of (B.2.5) is the greatest necessary value of t_{min}. The theorem follows. $\square$

In summary, both Lemma 10 and Theorem 6 provide a lower bound on t_{min} for the case that $t_{min} < t_i < 2t_{min}$ where $1 \leq i \leq n$. However, in contrast to Lemma 10, the lower bound of Theorem 6 guarantees that, if a packet of node i gets interfered by any node j, its next $2n-2$ packets will not be interfered again by node j. This result, together with Lemma 9, enables designing a reliable network, since it can be guaranteed that at least one packet of each node reaches its receiver in the worst case.

B.3. The bi-RARE protocol

Similar to the unidirectional protocol RARE, bi-RARE is based on the fact that many applications can tolerate data loss to some extent, in particular those that transmit their data periodically. To this end, to avoid the high costs of full reliability of (bi-)DEEP, bi-RARE is designed to allow for some data loss. That is, it can be configured for a probability $p < 1$ that data reaches its destination within a deadline in the worst case. The difference to RARE is that bi-RARE uses carrier sensing to reduce the chance of collisions and ACKs to decrease the number of packets on average.

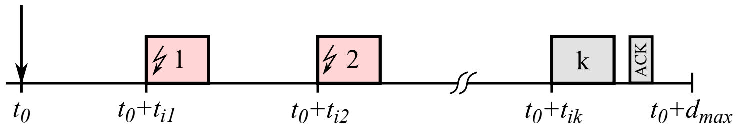


Figure B.5.: Basic working principle: After being activated by an event E, node i starts transmitting data packets in random time intervals until an acknowledgment is received — here for the k-th packet. This way, up to k packets are sent within a deadline d_{max} in the worst case.

Fig. B.5 depicts the working principle of bi-RARE in more details. Every time a node i is triggered by an event E, it starts transmitting redundant data packets of length L (see Fig. B.1) until an ACK is received or a maximum retransmission number k is reached — all within a deadline d_{max}. In contrast to (bi-)DEEP, inter-packet times are not constant, but node i waits a random time $t_{ix} \in \mathbb{R}_{>0}$ before sending any packet x — including the first one of a sequence. Here $1 \leq i \leq n$ and $1 \leq x \leq k$ hold and t_{ix} is uniformly selected from a time interval $[t_{min}, t_{max}]$ with $t_{min}, t_{max} \in \mathbb{R}_{>0}$. By waiting a random time before every transmission, packets are distributed equally over the available time (i.e., the deadline d_{max}). This balances traffic load and reduces collision numbers as shown later in simulation. Lastly, in contrast to (bi-)DEEP, there is no transmission pause after a sequence and nodes can be triggered directly anew after their deadline has passed, i.e., $t_{act} = d_{max}$.

While n, p, d_{max}, L and t_{act} are determined by the application and are known in advance, the remaining parameters k, t_{min} and t_{max} must still be selected. The following analysis stepwise derives these values, starting with a mathematical description of reliability, i.e., the probability that at least one out of k packets successfully reaches its destination. Now, to compute this probability, the worst-case transmission conditions need to be considered: (i) all n nodes in the network are sending (ii) there exists a maximum fraction of the interval $[t_{min}, t_{max}]$ for which any selected value of t_{ix} leads to a failed transmission. While condition (i) is straight forward, condition (ii) requires more analysis.

There are two possibilities that lead to a failed transmission attempt, either a collision or a blocked channel. Both possibilities can be expressed as time intervals, i.e., Δ_{col} for collisions and Δ_{blk} for a blocked channel. These describe the fraction of time in $[t_{min}, t_{max}]$ for which any selected value of t_{ix}, i.e., the point in time at which nodes start with carrier sensing, leads to either collision or blocking on the channel.

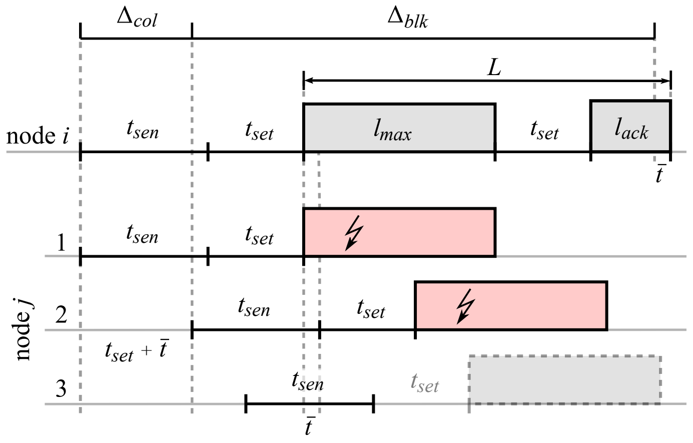


Figure B.6.: Illustration of the collision and blocking intervals Δ_{col} and Δ_{blk}. Whenever a node j starts carrier sensing in these intervals, its packet will either collide or be blocked by a packet of another node i. Case 1 and 2 show a collision between node i and j, whereas case 3 shows node j skipping its packet.

Regarding collisions, let us have a look at Fig. B.6. As depicted in case 1, a packet of node j collides with a packet of another node i, if both start carrier sensing at the same time. Now, if the packet of node j is further shifted to the right, i.e., it starts carrier sensing later, there will still be a collision as shown in case 2. Only when this shift is greater than $t_{set} + \bar{t}$, there will be no collision, but node j's packet will be skipped. That is, its carrier sensing interval now overlaps for more than $\bar{t}$ with node i's packet, which is sufficient for node j to detect it. As a result, the collision interval of a single packet can be described as:

$$\Delta_{col} = t_{set} + \bar{t}, \tag{B.3.1}$$

where t_{set} is the mode switch time and $\bar{t}$ the sensitivity of the transceiver.

The blocking interval Δ_{blk} can be determined similarly, as shown by case 3 in Fig. B.6. Here, a node j is able to detect a busy channel, if it starts carrier sensing more than $t_{set} + \bar{t}$ later than another node i, i.e., after the collision interval. This continues until the point in time where node j starts carrier sensing less than $\bar{t}$ time before node i's acknowledgment has finished. That is, the overlap of node j's t_{sen} with node i's acknowledgment is small enough and node j starts detecting a free channel. As a result, the blocking interval of a single packet can be described as follows:

$$\begin{aligned}\Delta_{blk} &= (t_{sen} - (t_{set} + \bar{t}) + t_{set} + L - \bar{t}) \\ &= (t_{sen} + L - 2\bar{t}) = t_{set} + L.\end{aligned} \tag{B.3.2}$$

Although there are differences between a packet failed due to a collision or a blocked channel regarding the consumed energy [66], it does not make any difference for reliability, since no packet will arrive at the sink in both cases. As a consequence, Δ_{col} and Δ_{blk} can be combined to describe the total fraction of time in $[t_{min}, t_{max}]$ for which any selected value of t_{ix} will lead to a failed transmission between any two single packets:

$$\begin{aligned}\Delta_{tot} &= \Delta_{col} + \Delta_{blk} \\ &= (t_{sen} + t_{set} + L - \bar{t}).\end{aligned} \tag{B.3.3}$$

In the following, t_{min} and t_{max} are derived, i.e., the bounds in which every node i uniformly selects its inter-packet times t_{ix}. To this end, t_{min} is set such that there can be at most one packet of each node in an interval of length $t_{max} - t_{min}$, for which it has to fulfill the following condition:

$$\begin{aligned}t_{min} &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max}}{2}.\end{aligned} \tag{B.3.4}$$

Note that t_{min} is the minimum and t_{max} the maximum separation between two consecutive transmission attempts of a node. If t_{min} is smaller than $\frac{t_{max}}{2}$, each node can send multiple packets within the interval $t_{max} - t_{min}$, for example, if it (randomly) selects t_{min} multiple times. This, however, leads to a lower worst-case performance and is therefore not meaningful, as shown in [63] and in the case of RARE in Section 6.3.2.

Given the fact that there can be at most one packet per node in $t_{max} - t_{min}$, it is possible to compute the maximum probability of packet loss for every packet being sent. This is the ratio between Δ_{tot} of all packets — if all nodes are transmitting, there can be $n - 1$ other packets that can cause interference — and $t_{max} - t_{min}$:

$$q = \frac{(n-1)\Delta_{tot}}{t_{max} - t_{min}}. \tag{B.3.5}$$

The probability of successful packet transmission in the worst case is given by $1 - q$. Note that for (B.3.5) to be valid the following condition must be satisfied (i.e., $q \leq 1$ must hold):

$$\begin{aligned}(n-1)\Delta_{tot} &\leq t_{max} - t_{min}, \\ t_{min} &\leq t_{max} - (n-1)\Delta_{tot}.\end{aligned} \tag{B.3.6}$$

Since network parameters, such as t_{min}, t_{max}, t_{sen}, etc., are common to all nodes, q is independent of the node and packet being sent. This allows modeling reliability, i.e., the probability p that at least one out of k transmission attempts reaches its destination for any node n, using a binomial distribution.

To this end, all possible combinations need to be considered, i.e., the first packet arrives, the second packet arrives, etc., which is a cumbersome procedure. To facilitate calculations, it is easier to compute $1-p$ instead, i.e., the probability that, in the worst case, no transmission attempt is successful. This is the probability that k consecutive packets are lost and can be computed by the well-known equation $\binom{k}{x}q^x(1-q)^{(k-x)}$ where $\binom{k}{x} = \frac{k!}{x!(k-x)!}$ is the binomial coefficient. Replacing q as per (B.3.5) and choosing $x = k$, i.e., k out of k packets are lost, the result is:

$$1 - p = \left(\frac{(n-1)\Delta_{tot}}{t_{max} - t_{min}}\right)^k. \tag{B.3.7}$$

For (B.3.7) to be valid, it has to be ensured that nodes are always able to send k packets within d_{max}. Towards this, recall again that every node i waits a random time t_{ix} chosen from $[t_{min}, t_{max}]$ before sending any packet. In the worst case, node i will select t_{max} for each of its k packets. To guarantee that even the last packet of node i has been transmitted before d_{max}, the following must hold:

$$t_{max} \leq \frac{d_{max} - (t_{sen} + t_{set} + L)}{k}. \tag{B.3.8}$$

Given a value of t_{max} as per (B.3.8), (B.3.7) can be reshaped to compute the value of t_{min} that satisfies a desired reliability p for the whole WSN:

$$t_{min} \leq t_{max} - \frac{(n-1)\Delta_{tot}}{\sqrt[k]{1-p}}. \tag{B.3.9}$$

It can be seen from (B.3.9) that full reliability, i.e., $p = 1$, is only possible for $n = 1$, independent of all other parameters. For $n > 1$, if p tends to 1, t_{min} tends to minus infinity as per (B.3.9). In other words, similar to RARE, 100% reliability as with TDMA or (bi-)DEEP cannot be achieved. However, bi-RARE allows for a reliability that is acceptably close to 100%, while considerably reducing the number of transmission attempts and, hence, making better use of energy. This is further discussed in simulation in Section 6.4.

B.4. Key findings

This chapter presented the bidirectional MAC protocols bi-DEEP and bi-RARE, which are extensions of DEEP and RARE with additional carrier sensing and acknowledgment mechanisms. The idea behind these protocols is to improve the average performance of the system by reducing the overhead of unidirectional communication, i.e., delays and packet numbers. More specifically, packet numbers can be reduced by using ACKs (a node can stop sending packets, if it knows that its data has been received) and collisions can be minimized by performing carrier sensing before every transmission (this can avoid simultaneous transmissions).

In the case of bi-DEEP, fully reliable communication can be guaranteed, i.e., data always reaches its destination in the worst case. To this end, similar to DEEP, each node transmits its data as sequences of redundant packets with constant inter-packet times. However, in contrast to DEEP, inter-packet times and packet numbers are calculated differently due to the additional carrier sensing and ACK mechanisms. That is, inter-packet times do not depend on the packet length anymore, but only on the transceiver quality, i.e., how fast it can switch modes (t_{set}) and how sensitive it is ($\bar{t}$). This typically results in shorter inter-packet times, in particular, if fast hardware is used or packet sizes are large. On the other hand, each node now needs to transmit $k = 2n - 1$ packets per sequence instead of $k = n$, which is almost twice as much. A detailed evaluation and comparison to DEEP can be found in Section 6.4.6.

The bi-RARE protocol makes use of the fact that many applications can tolerate some data loss and do not require full reliability. To this end, it can be configured to a user-specified reliability of $< 100\,\%$, i.e., to a certain probability that data reaches its destination in the worst case. Similar to RARE, nodes transmit sequences of k packets with inter-packet times that are randomly selected from an interval $[t_{min}, t_{max}]$. However, calculations vary slightly leading to different performance. This is further evaluated in Section 6.4.6.

In summary, bi-DEEP and bi-RARE can increase the average performance of a system, i.e., reduce packet numbers, delays and energy consumption. However, there are also several drawbacks compared to DEEP and RARE. In particular, complexity is much higher, which generally leads to a lower worst case performance of system, since both carrier sensing and ACK introduce two additional error sources. Further, if slow transceivers are used or packet numbers are short compared to carrier sensing or ACKs, the benefits of bidirectional communication vanishes. This is analyzed more closely in Section 6.4.6, where bi-DEEP and bi-RARE are evaluated and compared to other protocols in a series of simulation-based experiments.

Summary of findings

1. Under negligible external interference, it is possible to guarantee fully reliable communication in unidirectional networks by carefully selecting packet numbers and inter-packet times. This resulted in the proposed DEEP algorithm of Chapter 4.

2. On the other hand, an arbitrarily high reliability (less than 100 %) is also possible in unidirectional networks by randomly selecting inter-packet times within configurable boundaries. This has led to the proposed RARE protocol of Chapter 5.

3. Overall, RARE leads to less energy consumption and less transmission delay than DEEP. However, in contrast to DEEP, RARE can never achieve 100 % reliability.

4. Using feedback mechanisms such as acknowledgments and carrier sensing (i.e., bidirectional communication) always deteriorates the achievable reliability in the worst case. On average, bidirectional protocols are more beneficial, if network traffic is comparatively low.

5. Unidirectional communication is more robust against external interference than bidirectional communication, since nodes are active for a shorter time and there are less sources of errors such as acknowledgments, carrier sensing, etc. that can be affected by external transmissions.