Polymorphic network protocol suite in heterogeneous wake-up IoT networks

Sebastian Lucas Sampayo

UNIVERSITÉ DE STRASBOURG

ÉCOLE DOCTORALE MATHÉMATIQUES, SCIENCES DE L'INFORMATION ET DE L'INGÉNIEUR

Laboratoire ICube – UMR 7357

THÈSE présentée par :

[Sebastian Lucas SAMPAYO]

soutenue le : **17 mars 2021**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**

Discipline/ Spécialité : Informatique

Polymorphic network protocol suite in heterogeneous wake-up IoT networks

THÈSE dirigée par :

M. NOEL Thomas	Professeur, Université de Strasbourg
M. MONTAVONT Julien	Maître de conférences, Université de Strasbourg

RAPPORTEURS :

M. BERDER Olivier	Professeur, Université de Rennes 1
M. GALLAIS Antoine	Professeur, Université Polytechnique Hauts-de-France

---

AUTRES MEMBRES DU JURY :

M. VALOIS Fabrice	Professeur, INSA Lyon

Sebastian Lucas SAMPAYO

Polymorphic network protocol suite in heterogeneous wake-up IoT networks

Résumé

L'Internet des objets et les réseaux de capteurs sans fil offrent de nouvelles façons de connecter des entités physiques du monde réel au monde cybernétique. Ceci est réalisé grâce à des systèmes embarqués distribués, alimentés par des batteries. Le principal défi dans ces applications est de prolonger autant que possible la durée de vie de la batterie, ce qui se traduit par une minimisation de la consommation d'énergie, tout en conservant des performances réseau de bonne qualité. Dans ces systèmes, le module de communication sans fil est généralement celui qui consomme le plus d'énergie. En conséquence, les solutions traditionnelles à ce défi ont utilisé une approche de « duty-cycle » au niveau de la couche de contrôle d'accès au support (MAC) de la pile de communication en sacrifiant la latence au profit de l'efficacité énergétique. Les protocoles asynchrones permettent des communications non programmées, et leur mise en œuvre est facile à déployer et à exploiter. Toutefois, dans plusieurs cas, les protocoles synchrones offrent de meilleures performances. La « wake-up radio » est une nouvelle technologie pour les communications sans fil qui permet de conserver des communications asynchrones tout en favorisant la latence et la consommation énergétique. Ce module est attaché à un nœud régulier comme récepteur secondaire qui écoute le canal en permanence pendant que la radio principale reste en veille. Malheureusement, la sensibilité de la « wake-up radio » est inférieure à celle de la radio principale, ce qui crée un problème de discordance de portée. L'objectif de ce travail est de proposer de nouveaux protocoles de communication tirant parti de cette nouvelle technologie, afin de conserver la simplicité d'une approche asynchrone tant en proposant de performances similaires aux approches synchrones.

Résumé en anglais

The Internet of Things and the Wireless Sensor Networks are providing new ways to connect physical entities from the environment to the cybernetic world. This is made possible thanks to distributed embedded systems powered by batteries. The main challenge in these applications is to extend the battery lifetime as much as possible, which translates into minimizing the power consumption while keeping good quality network performance. In such systems, the wireless communication module is typically the most power-consuming one. As a consequence, traditional solutions to this challenge have used a duty-cycle approach at the medium access control layer (MAC) of the communication stack trading off latency for energy efficiency. Asynchronous protocols allow unscheduled communications, and their implementation is easy to deploy and operate. However, in many cases, synchronous protocols provide better performance. Wake-up radios are a new technology for wireless communications that allows holding asynchronous communications while favoring latency and energy consumption. This module is attached to a regular node as a secondary receiver that listens to the channel continuously while the main radio stays sleeping. Unfortunately, the sensitivity of the wake-up receiver is lower than that of the main radio, creating a range mismatch problem. The goal of this work is to propose new communication protocols taking advantage of this technology, aiming at keeping the low complexity of an asynchronous approach while allowing similar performance to that of a synchronous approach.

Université de Strasbourg

École doctorale MSII

Polymorphic network protocol suite in heterogeneous wake-up IoT networks

THÈSE

pour obtenir le grade de

Doctorat de l'Université de Strasbourg

mention Informatique

17 mars 2021

présentée par

Sebastian Lucas SAMPAYO

Composition du jury

Directeur de thèse:	Prof. Thomas NOEL, Université de Strasbourg, France
Co-encadrant:	Prof. As. Julien MONTAVONT, Université de Strasbourg, France
Rapporteurs:	Prof. Olivier BERDER, Université de Rennes 1, France Prof. Antoine GALLAIS, Université Polytechnique Hauts-de-France, France
Examinateurs:	Prof. Fabrice VALOIS, INSA Lyon, France

Laboratoire des sciences de l'ingénieur, de l'informatique et de l'imagerie
ICUBE — UMR 7357

Acknowledgments

First of all, I would like to thank my research supervisors Pr. As. Julien Montavont and Pr. Thomas Noël for having trusted me and facilitated the development of this project, for their helpful advice, from whom I was able to learn the good practices of research.

Also, thanks to the members of the ANR WakeUp project from the University of Rennes, the CEA-LETI, and Wi6Labs, with whom I could exchange ideas, and grow professionally in the framework of a multi-disciplinary and multi-regional project.

In like manner, I would like to thank the permanent researchers of the Networks team at the ICube laboratory who always accompanied my path positively.

Moreover, I would like to kindly thank Pr. Olivier Berder, Pr. Antoine Gallais, and Pr. Fabrice Valois for accepting to be the members of the jury.

A big thanks to Julián who helped me enormously in my arrival and stay in Strasbourg, as well as the rest of the doctoral, master, and engineering students of ICube, with whom I had fun, shared the stress, learned and collaborated.

I extend my deep gratitude and respect to SATT Conectus for trusting my project and my profile to give a new perspective to my Ph.D. for the future.

Furthermore, I would also like to acknowledge the University of Buenos Aires, for giving me a high-level theoretical background that has allowed me to go deeper into various research domains.

I am particularly grateful to Chantal and Guy, and their family and friends, for opening me the doors of French culture and traditions, and above all, for supporting me and my wife and making us feel at home, like family.

A special thanks to my parents for having given me the tools to achieve whatever I set out to do, as well as for having supported and encouraged me in every decision of this project.

Finally, I am thankful to my wife, Antonella, for her unconditional love, understanding, and joy, without which it would not have been possible to go so far.

Abstract

Every year, more devices are getting connected to the Internet in different life domains such as Smart Buildings and Smart Transportation. The global Smart City market was valued at five hundred billion dollars in 2017 and is projected to reach two thousand billion by 2025 [1]. Wireless Sensor Networks (WSN) are commonly used for such applications where there is a need for measuring some physical variable of the environment, and make the information available on the Internet. The nodes for this goal comprise low power and resource-constrained devices with a limited distance range of communication. To cover wider areas, these nodes interconnect wirelessly in what is called multi-hop WSN. The main challenge in these applications is to extend the battery lifetime as much as possible, which translates into minimizing the power consumption while keeping good quality network performance.

Traditionally, the energy consumption was controlled in these networks by some form of duty-cycle in the communication protocol at the MAC layer trading off latency for energy efficiency [2]. In recent years, the Wake-Up Radio (WuR) technology has advanced with increasing acceptance because it promises the end of this tradeoff [3]. The WuR is a secondary receiver that listens to the channel continuously while the main transceiver stays sleeping and only wakes up on-demand by a signal on the WuR channel. Then, the data is transmitted using the main radio. The essentials of it are explained in Chapter 2.

The goal of this thesis is to design a new protocol stack benefiting from wake-up radio technology to keep the advantages of an asynchronous protocol (such as low complexity, easy deployment, and operation) together with the performance of a synchronous protocol (high throughput and predictable reliability). Part I of this thesis presents our investigation on wake-up radios at the MAC layer together with our first main contribution on a new MAC protocol based on wake-up radios. We also present the simulations and experiments on a real prototype that we performed to evaluate it. Next, we analyze in Part II the impact of this technology at the routing layer. The main problem is that the range of WuR is shorter than that of the main radio. Consequently, the use of WuR leads to very dense networks potentially increasing the number of hops that are required to communicate a data packet. With that in mind, we design REFLOOD, a reactive routing protocol, addressing the range mismatch problem.

List of Publications

International Journals

1. S. L. Sampayo, J. Montavont and T. Noel. REFLOOD: Reactive Routing Protocol for Wake-Up Radio in IoT, submitted to Elsevier Ad Hoc Networks (pending review).

International Conferences

1. S. L. Sampayo, J. Montavont and T. Noel. eLoBaPS: Towards Energy Load Balancing with Wake-Up Radios for IoT, in the 18th International Conference on Ad Hoc Networks and Wireless (AdHoc-Now), Luxembourg, Luxembourg, October 2019, rank B.

2. S. L. Sampayo, J. Montavont and T. Noel. LoBaPS: load balancing parent selection for RPL using wake-up radios, in the 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, July 2019, rank B.

3. S. L. Sampayo, J. Montavont, F. Prégaldiny and T. Noel. Is Wake-Up Radio the Ultimate Solution to the Latency-Energy Tradeoff in Multi-hop Wireless Sensor Networks?, in the 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Limassol, Cyprus, October 2018, rank B.

Workshops

1. S. L. Sampayo, J. Montavont and T. Noel. A Performance Study of the Behavior of the Wake-Up Radio in Real-World Noisy Environments, in AWAKE workshop of the 2020 International Conference on Embedded Wireless Systems and Networks on Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks (EWSN), Lyon, France, February 2020.

National Conferences

1. S. L. Sampayo, J. Montavont and T. Noel. Selecting Parents with Wake-Up Radios for Load Balancing in RPL, in the 4ème rencontres francophones sur la conception de protocoles, l'evaluation de performance et l'expérimentation des réseaux de communication (CoRes), Saint Laurent de la Cabrerisse, France, June 2019.

Contents

Introduction and background

Chapter 1

Introduction

Contents

The research conducted in this thesis took place between February 2018 and January 2021. Our primary focus was the design of communication protocols for a new wireless technology of the Internet of Things (IoT): Wake-up Radios (WuR). As stated by the title, it is a protocol *suite* because it comprises manifold layers of the network stack. It is *polymorphic* because they have a hybrid behavior that may change according to the current status of the network. Finally, it is *heterogeneous* because they are made for IoT devices that use multiple wireless radio technologies simultaneously. This work is part of the WakeUp project, founded by the French National Research Agency (ANR), where we collaborated with an academic partner (University of Rennes 1), a research institution (CEA-LETI), and a startup (Wi6Labs).

In Wireless Sensor Networks (WSN), the main goal is to reduce power consumption as much as possible. Traditionally, the solution for this is to use duty-cycle mechanisms at the Medium Access Control (MAC) layer. However, there is a tradeoff because if the power consumption is reduced a lot, then the latency becomes high. Many applications in the industrial domain require both energy efficiency and low latency, for example in control loops with sensors and actuators. Wake-up radios technology promises the end of this tradeoff by reducing both magnitudes at the same time.

Most of the work that has been done so far towards the development of wake-up radios has concerned the hardware side, and not too much attention has been paid to the networking counterpart.

1.1 Context

In the following sections, we provide a general background to understand the origin of this thesis.

1.1.1 The Internet of Things

Many authors in the past few years have agreed on the fact that the world is transitioning to a new industrial revolution [4], [5]. Not only because of the digital drivers such as Artificial Intelligence (AI) and Internet of Things (IoT), but also because of the biotechnologies that are enabling synthetic biology, and gene sequencing, to name a few. The fourth industrial revolution is transforming people's lives through new ways of working and interacting with one another, as humankind has never seen before. The speed of these changes is now exponential instead of linear compared to the previous revolution. The evolution is so deep that it is changing who we are through a paradigm shift, from an information and communication society into a super-intelligent society [5].

The agrarian revolution, 10.000 years ago, with the domestication of animals, allowed people to move from foraging to farming. This changed profoundly the society that moved from a nomad style to larger settlements, and the beginning of urbanization and cities. The first industrial revolution, between the 18th and 19th centuries, driven by the railroads and the steam machine, produced a shift from manual to mechanical production. A few decades later, at the end of the 19th century, the second industrial revolution enabled mass production thanks to electricity and assembly lines. In the decade 1960, the third industrial revolution has started with the MOS transistor invention, leading to computers and the Internet.

Nowadays, since the beginning of the 21st century, the fourth industrial revolution has been characterized by the broad spread of the Internet, not limited to terminal computers, but connecting every object from the physical world into the digital realm. Objects that were only part of the physical environment in the past century and could only interact with humans manually are now equipped with a piece of hardware running a software program, with connectivity to the cybernetic world. These objects now can interact between them, in what is so-called, *device-to-device* or *machine-to-machine* communication, to further automate processes in our daily lives. In the beginning, this connectivity has been wired with the same infrastructure as the initial Internet. However, with the increasing applications of cyber-physical systems, wireless communication systems are giving more flexible ways to provide ubiquitous and mobile connectivity to everyday things. That is the definition of the Internet of Things (IoT). The evolution of this path of humankind industrialization is depicted in Fig. 2.

A particular subdomain of the IoT is that of the Wireless Sensor Networks (WSN). The nodes in these networks comprise low power and resource-constrained embedded systems with wireless communication capabilities.

1.1.2 Wireless sensor networks

The goal of WSN in IoT is to measure physical variables from the environment and make them available on the Internet. Additionally, commands can be sent from the

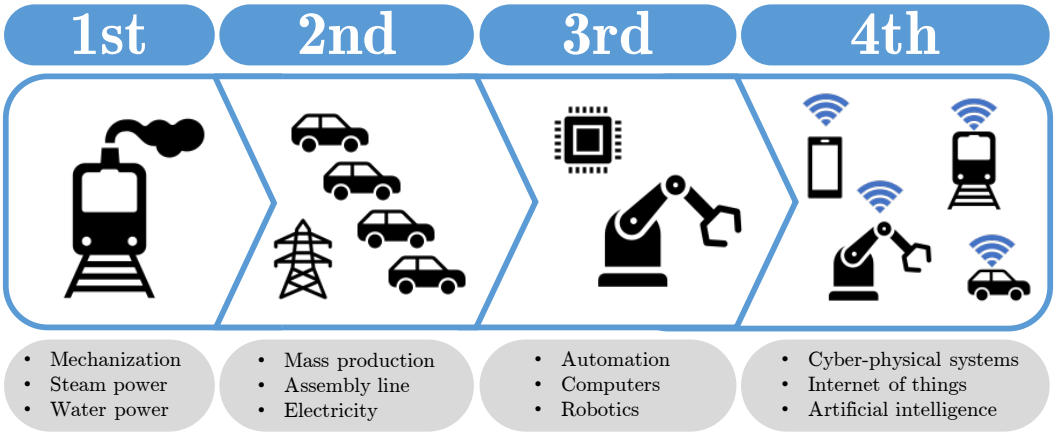


Figure 1.1: Industrial Revolutions along history

Internet towards actuators to modify the physical environment.

In WSN, the nodes are small devices with limited power and computation capabilities. Each node is equipped with one or multiple sensors that measure some physical variable of the environment. Using a simple microcontroller, the device can process that information and with the help of some wireless radio, it conveys the data to a special node, called *sink* or *gateway*. The sink is in charge of collecting the data from all the sensors in the network, process it, and possibly connect it to the Internet. The traffic pattern that is generated for this goal is commonly called *convergecast* or *multi-point-to-point*. This is the main pattern that we will address in this thesis.

The fact that the resources of these devices are constrained, translates into a limited range of communication. Then, to cover a wider area, the nodes interconnect between them in what is called *multi-hop* network. Intermediate nodes, in that case, relay data packets from those that are further away.

There are many examples around this idea, such as the Smart City, Smart Factory or Smart Building [6]. There are many tasks in a building that are traditionally performed manually, with human work. From the simplest one, like turning on and off the lights with a switch, up to more complex ones, like detecting a water leak or structural damage. Collecting data through WSN allows us to be proactive, i.e. take actions before the occurrence of such an event. This reduces maintaining costs and allows the use of the resources more efficiently. Furthermore, we can also combine the measurements with artificial intelligence algorithms to predict what is going to happen in the future and take better decisions accordingly.

One of the main challenges in WSN in the upcoming years is to extend the battery lifetime of the network, while maintaining a good quality of communication, fulfilling the application requirements. The reason is that the frequent manual replacement of the batteries of thousands of devices is not feasible [7], [8], [9].

1.2 Motivations

Every year, in the whole world, more than 15 billion batteries are thrown away, containing toxic materials that contaminate the environment, and create serious health problems to the population. Consequently, this increases the medical costs associated to heal these issues. Furthermore, large deployments may consist of up to ten thousand sensor devices, where the logistical costs of battery replacements are huge. This is a huge blocker for the adoption of new technologies that can improve significantly the efficiency in many industries. For this reason, we need to reduce and ultimately eliminate the use of batteries.

Traditionally, the most energy-consuming sub-system of a sensor node is the transceiver module [7]. Hence, the energy consumption is reduced in these devices by duty-cycling the activity at the Medium Access Control (MAC) layer of the network stack [7]. This way, the node is only able to communicate in a short active period, while most of the time the transceiver is in sleep mode. The communication with such devices can be achieved *synchronously*, by coordinating the transmission and reception timeslots for each node, or *asynchronously*, where the nodes do not agree on any special schedule. Synchronous MAC protocols have proven to be very efficient in high traffic scenarios and stable networks [8]. However, in many WSN applications, the traffic load is low and the network is dynamic. In those cases, the synchronous protocols present a lot of overhead and their implementations, deployments, and operations remain difficult. Asynchronous MAC protocols are better suited for those applications because they do not keep track of any schedule and make no assumptions about the network. As a consequence, the device has to check the channel periodically introducing idle listening (when there is no other node that needs to communicate) or overhearing (when the device is not the intended destination of the communication) phenomena that can significantly impact the battery consumption. Additionally, collisions and protocol control overhead waste energy in transmissions that we want to reduce as much as possible. Finally, the duration of the active period results in a tradeoff between latency and energy efficiency in these types of solutions, because of the limited time to communicate in each cycle.

Recently, Wake-up Radios (WuR) technology has advanced as a new solution to extend the network lifetime in WSN [3]. Its ultra-low power consumption and always-on feature overcome the problems of traditional duty-cycled asynchronous MAC protocols. The WuR is a secondary receiver that listens to the channel continuously while the main transceiver stays sleeping. When a node wants to communicate a data packet, it first sends a wake-up signal (a packet on the WuR channel) towards the destination. Upon reception, the destination wakes up the main radio and waits in listening mode for the data packet. After exchanging data and acknowledgment, the destination puts back its main radio into sleep mode to save energy. A detailed introduction to WuR is presented in Section 2. This technology is revolutionary because it promises the end of the idle listening and overhearing at the MAC layer of asynchronous protocols, and no more precise synchronized algorithms.

One of the main limitations of WuR is a low sensitivity, which translates into a range mismatch between the main radio (long or medium range, in the order of 200 m) and the WuR (short-range, around 20 m) [9]. Consequently, the use of WuR leads to very dense networks, because the nodes must be close to each other,

to communicate through the very short range of WuR. As a result, the network becomes a two-tier architecture in which we have to deal with two sets of neighbors (one for each radio interface) that potentially overlap. This problem is crucial to spread the wake-up radio technology, and it has not been studied extensively yet.

Therefore, we highlight the following research challenge in this thesis:

Scientific challenge

Can we achieve with WuR-based asynchronous protocols the same level of performance as that of synchronous ones, with all the benefits of the former ones? How to deal with such a two-tier architecture both at the MAC and network layer?

1.3 Goal and main contributions

The main goal of this thesis is to explore the benefits of the wake-up radio technology at the MAC layer and the routing layer, in WSN applications. The main benefit of WuR is at the MAC layer because it eliminates duty-cycle and reduces both latency and power consumption at the same time. It enables pure-asynchronous protocols, which are easier to implement, deploy, and operate them than traditional synchronous one, and with the same level of performance. Regarding the OSI model, the routing layer should be agnostic of the underlying technology used in lower layers. However, the WuR introduces a secondary network, in parallel with the primary one, with its own links. Calculating routes over the multi-graph created by main radio and WuR connectivity is a difficult task and has not been studied yet. In particular, we are interested in the network performance of the wireless communication protocols in these applications. We address the main challenge of WSN: extend the lifetime of the network as much as possible, while keeping high levels of reliability and low latency. To this end, we analyze the MAC and routing layer, and also perform experiments at the physical layer. Based on those analyses, we designed new MAC and routing protocols that benefit from the best of the WuR technology. In this thesis, we present two main contributions: the analysis at the MAC layer and the new solution at the routing layer.

In our first contribution, we investigate the benefits, drawbacks, and tradeoffs of using WuR in multi-hop WSN in terms of the packet delivery ratio, latency, and power consumption at the MAC layer. Our analysis is based on evaluations using COOJA [10], a simulator for networks of ContikiOS nodes [11]. Our findings show that there is a threshold in the size of the network for WuR to perform efficiently. Increasing the network size beyond this threshold significantly degrades the WuR performance, making a traditional duty-cycled MAC protocol a better choice for such configuration. We also show that acknowledging the wake-up signal is problematic in the presence of collisions because it decreases seriously the reliability of the network. Then, we perform a study on the behavior of a WuR prototype when

it is subject to a real-world noisy environment. We analyze how interference can be wrongly considered as valid packets and how to deal with them. We show the importance of the utilization of the Clear Channel Assessment (CCA) capabilities to reduce transmission errors, resulting in a higher packet delivery ratio. Besides, we extract some key physical values of the prototype that can serve in modeling WuR communications. Finally, we provide a method to estimate the overall current consumption of an application deployed over a WuR-based network. The results show that radio communications account for a negligible part of the energy depletion in low traffic scenarios, meaning that further optimizations in the communication protocol stack will not improve the lifetime of end-devices in such cases.

The second contribution goes beyond the MAC layer to analyze the impact of using WuR at the routing layer and explore its benefits. There, we present LoBaPS, a cross-layer approach that relies on a pre-established routing structure and combines the best of two worlds: the power efficiency and always-on feature of WuR with the stability of a well-known configuration of the Routing Protocol for Low Power and Lossy Networks (RPL) [12]. Moreover, we put the focus on load balancing to extend the lifetime of the network. This metric, together with the latency and packet delivery ratio, where extracted from simulations. The results show the robustness of the solution because the network can adapt quickly to the shutdowns of nodes. Encouraged by those results, we extended LoBaPS to focus on energy in a new solution referred to as Energy LoBaPS (eLoBaPS). The energy savings achieved are reflected in the resulting lifetime that is compared to that of a reference WuR-based protocol and LoBaPS. Besides, we look at the packet delivery ratio of the network over time to compare the stability and final decline of the operation performance, where eLoBaPS has a longer stable operation and a shorter decline. Furthermore, we present a new metric called *productivity* that reflects both reliability and lifetime independently of the traffic scenario. By improving the load balancing towards the ideal case, our protocol extends the network lifetime up to 77%. Besides, the network behavior becomes more stable over its lifetime and the decline with degraded performance is shorter. eLoBaPS presents many benefits but requires a routing structure to operate. Its performance directly depends on the solution used to build and maintain the routing structure. Then, we decided to remove this dependency by designing a self-contained protocol referred to as REFLOOD. It is a reactive routing protocol that leverages the characteristics of WuR. We compare REFLOOD to a proactive approach via an exhaustive series of simulations performed in ContikiOS/COOJA. The results show that REFLOOD represents an improvement to 300% of the network lifetime of the traditional solutions. Its multiple-paths feature maximizes the chances to wake-up a destination successfully, improving the packet delivery ratio, and reducing the latency. Furthermore, we showed that it is more robust to topology changes. Finally, the lack of control overhead in reactive protocols is especially important to increase the network lifetime in low traffic applications with wake-up radios.

1.4 Structure of the thesis

This thesis is organized into two parts: Part I for the MAC layer analysis and Part II for the routing layer solution. Each part contains a chapter for every single

contribution and a conclusions chapter with a summary of the set of contributions.

Chapter 2 describes the wake-up radio technology in detail, providing the foundations needed to follow the next chapters. Inside Part I, Chapter 3 explores the benefits of WuR at the MAC layer, while Chapter 4 describes the experiments we carried on to study the behavior of the WuR technology in a noisy environment with a real prototype. At the end of Part I, we present a summary of the MAC layer analysis. Moving into Part II, Chapter 5 describes an introduction to the challenges of using WuR from a routing perspective and presents LoBaPS, a cross-layer solution with a focus on load balancing. Afterward, in Chapter 6, we present REFLOOD, a self-contained reactive routing protocol leveraging WuR. To conclude Part II, we provide a summary of these contributions. Finally, Chapter 7 concludes this research on communication protocols for wake-up radio technologies and proposes some potential research directions.

Chapter 2

Wake-up Radio state-of-the-art

Contents

In this chapter, we describe the WuR technology, providing details on the physical layer, the most common system architecture, the challenges that hinder its spread, and finally a brief review of the existing solutions.

2.1 Introduction

The conception of an ultra-low-power transceiver started back in the 2000 decade with the *PicoRadio* project at the University of California, Berkley, Wireless Research Center, guided by Pr. Jan Rabaey. There, they aimed at designing *PicoNodes*, that is, nodes that are smaller than 1 cm^3, weigh less than 100 g, cost less than $ 1, and consume less than 0.5 mW. This way, the energy required to operate can be harvested from the environment avoiding the problem of recurrent battery replacement. To reduce the power consumption that much, it is necessary to optimize every single component of the system. Traditionally, the subsystem that consumes the most in WSN applications is the radio module. Therefore, a lot of research efforts have been made towards an ultra-low-power receiver, called Wake-up Radio, whose power consumption is more than two orders of magnitude less than that of other wireless radio technologies [3].

Over the past few years, great progress has been made towards the design of the hardware system, together with improved capabilities of new semiconductor processes. However, not so much attention has been paid to the software side and the network performance. The wireless communication protocols play an important role

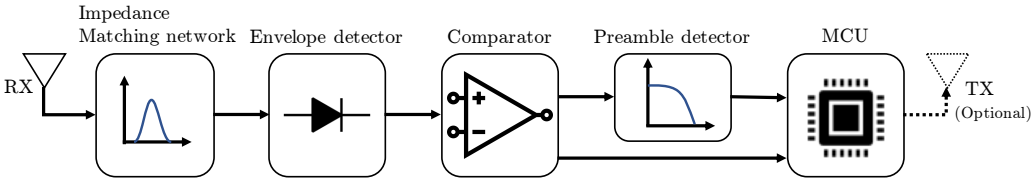


Figure 2.1: WuR blocks diagram

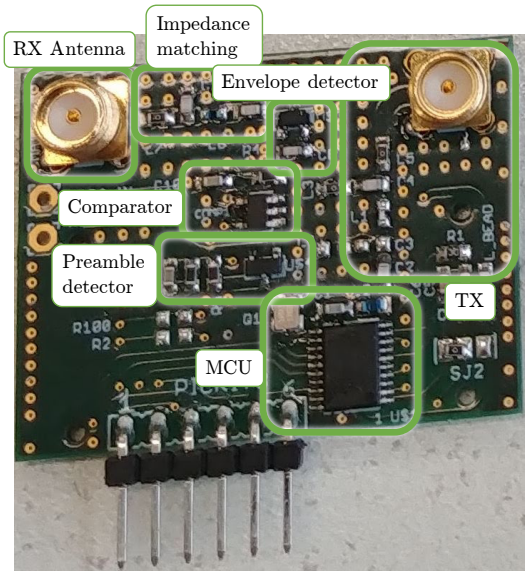


Figure 2.2: WuR prototype used in this thesis

to make sure that the optimal power consumption levels are achieved. Furthermore, in large deployments, reducing the power consumption of a single device is not enough. It is necessary to optimize the power consumption and overall performance of the network as a whole to make sure that the high-level-application requirements are met. This can determine the success or failure of this technology.

2.2 Physical layer

As the fabrication technology of CMOS devices advances towards smaller sizes, the designs of the WuR receiver has been reducing its power consumption, improving sensitivity, and wake-up latency. In general, the communication system for this module uses On-Off-Keying (OOK) modulation. This scheme consists of simply turning on and off the carrier frequency to transmit a digital 1 or a 0 correspondingly. There is not any consensus on the selection of the carrier frequency. On the one hand, it depends on the unlicensed bands' availability. On the other hand, reducing the frequency improves the effective range but increases the size of the antenna. Typically, the circuits required to implement OOK consume less power than other modulation schemes. Many designs for the receiver are based on the heterodyning technique, but also passive rectifiers have been used since they prove to be very efficient. Normally, there is a tradeoff between energy and sensitivity between both

techniques. Furthermore, some heterodyne designs use duty-cycle techniques that propose a tradeoff between energy and latency, though with lower levels than those of traditional radios.

Some of the state-of-the-art designs are compared in Table 2.1. One of the first wake-up receivers has been designed by Nathan M. Pletcher in 2008 [13]. The prototype, implemented in 90 nm CMOS technology, achieved a sensitivity of -72 dBm at 100 kbps with a power consumption of 52 uW at 2 GHz, based on a heterodyne receiver architecture. In 2016, Camilo Salazar presented another design at 2.4 GHz, in this case, implemented with 65 nm CMOS technology, and achieving -97 dBm sensitivity at 10 kbps, while consuming 99 uW [14]. Recently, Anjana Dissanayake presented a new design in 65 nm CMOS, that achieves a sensitivity of -99 dBm at 434 MHz [15]. A flexible configuration, in that case, provides an option that consumes only 260 nW but a wake-up latency of 2.6 s, and another option consuming 2.17 uW with a latency of 260 ms.

On the other side, many authors have presented wake-up receivers with discrete off-the-shelf components [3], reducing the time and cost of prototype development. In general, these proposals are based on an energy detector instead of a coherent receiver because the power consumption is extremely reduced. However, the main drawback is that the sensitivity is poor compared to other architectures. Michele Magno presented in 2016 a prototype that consumes 1.2 uW and achieves the best sensitivity in this type of designs: -55 dBm at 868 MHz with a wake-up latency of only 8 us [16]. Another configuration of the same prototype offers lower power consumption (152 nW) at the cost of -32 dBm of sensitivity. The power consumption and wake-up latency of these designs are the lowest in the literature. In this thesis, we have made experiments and tuned our simulations based on that prototype, which is illustrated in Fig. 2.2.

The block diagram of that design is shown in Fig. 2.1. It consists of an initial *impedance matching filter* that maximizes the power transfer from the antenna to the circuit. Then, an *envelope detector* provides an output voltage proportional to the received signal energy, demodulating it into OOK. Afterward, the signal is compared to an *adaptive threshold* that is self-adjusted to be in the middle of the incoming signal amplitude. The resulting digital signal is then fed into an *ultra-low-power microcontroller* that processes the content of the wake-up signal. This is typically an 8-bits microcontroller. In parallel, the output of the comparator is used in the *preamble detector* to filter high-frequency noise in the channel. The output of that sub-system triggers an external interrupt of the WuR microcontroller which uses this information to validate the data received from the comparator. The microcontroller used in this prototype includes a transmitter that can be used to transmit frames to other wake-up receivers. We are going to call these frames hereafter, Wake-up Signals (WuS).

2.3 System architecture

The main goal of an IoT node for WSN is to measure the physical world, digitalize it and process it, and then communicate it wirelessly to other devices to ultimately reach the Internet. An IoT node is implemented with an embedded system that combines electronic components with a software layer. An example of this can be

Table 2.1: Wake-up receiver designs

Authors	Year	Power [uW]	Sensitivity [dBm]	Technology	Frequency [GHz]
Pletcher	2008	52	-72	IC 90nm	2
Salazar	2016	99	-97	IC 65nm	2.4
Dissanayake	2020	0.260/2.17	-99	IC 65nm	0.434
Magno	2016	1.2/0.152	-55/-32	PCB	0.868

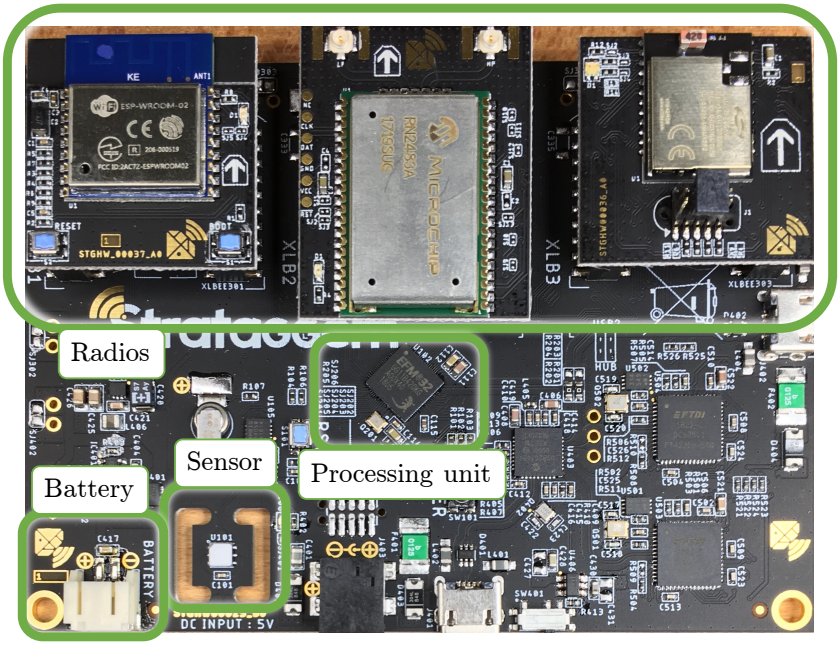


Figure 2.3: Example of an IoT node with multiple radios, developed by the company Strataggem, model STGHW00025

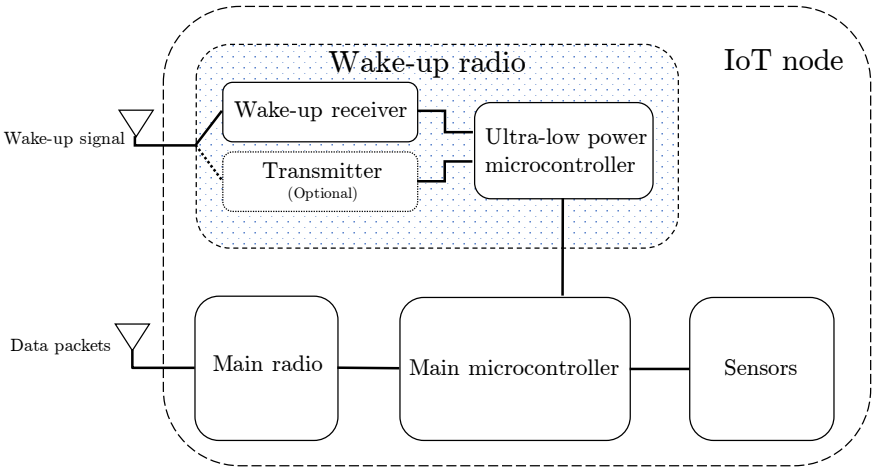


Figure 2.4: Architecture of an IoT node with Wake-up Radio

seen in Fig. 2.3. This electronic board contains a sensor that measures temperature and humidity from the environment and converts it into a digital signal. This information is further processed in the main microcontroller (MCU), in this case, one of the EFM32 family from Silicon Labs, based on an ARM Cortex M4 architecture. Then, one or multiple radio modules are used to communicate the data wirelessly.

In the applications we are concerned about in this thesis, we will consider two radio modules: a main and a secondary one. The main radio module can be any existing technology that is used to exchange data with other nodes, such as IEEE 802.15.4 radios, WiFi, or Bluetooth. The secondary radio is the WuR module. Its key feature is that the ultra-low-power receiver remains always-on listening to the WuR channel. In some cases, the wake-up signal can be transmitted by the main radio if it can generate the required modulation and power consumption targets. In other cases, the WuR module contains optionally a low power transmitter, as depicted in Fig. 2.4. This is the case for the prototype we have used. The main goal of this architecture is to put to sleep the main radio and only wake it up when there are incoming or outcoming data to communicate.

This way, when a node wants to communicate a data frame, it first sends a signal on the WuR channel towards the destination. Upon reception, the destination wakes up its main radio and waits in listening mode for the data frame. After exchanging data and acknowledgment, the destination puts back its main radio into sleep mode to save energy. This is the simplest and most direct way of communicating at the MAC layer using WuR. An implementation of this protocol is found in [17], called W-MAC. We are going to use it as a reference multiple times throughout this thesis.

A WuS can be a single beam that just works as a trigger for the wake-up receiver. Additionally, it can be modulated with digital data to convey more information that is decoded by the comparator and the microcontroller submodules. Typically, the information embedded in the WuS is an identifier of the destination, so that other nodes receiving this signal ignore it and continue sleeping. Otherwise, nodes that are not intended to receive the WuS would wake up their main radio and overhear a data frame. In the literature, this is known as a *false wake-up*. In this work, we will take advantage of this feature to avoid such a phenomenon. Moreover, given the flexibility of a microcontroller in the WuR module, we can put more information in the wake-up signal to enable new mechanisms in the control plane of the network protocols, as we are going to see in the next chapters. The communication between the microcontroller of the WuR module and the main microcontroller of the IoT node is typically SPI or I2C plus an interrupt signal. This allows the main microcontroller to send configuration commands to change the behavior of the WuR module, for example, to modify the address.

Theoretically, the WuR could be used to send the data frames directly, without the main radio. Nevertheless, OOK modulation is more prone to interferences and collisions, and the low data rate (from 1 kbps to 100 kbps) translates into long times over the air for each frame. Comparatively, the main radio typically provides more robust modulation techniques and higher data rates.

Provided that the hardware components of the WuR circuit are chosen carefully, the operating data rate can be modified by firmware. However, increasing the data rate too much reduces sensitivity [18].

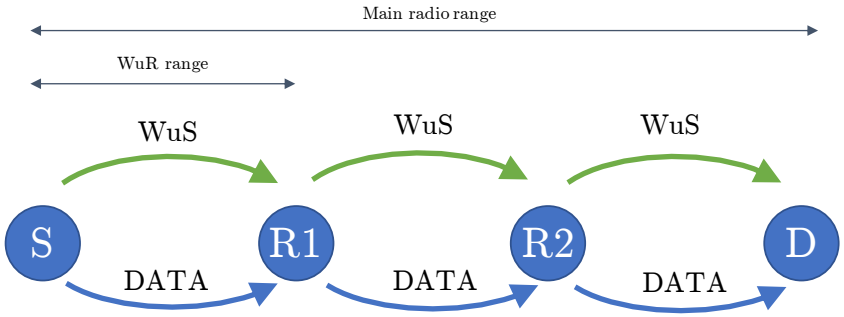


Figure 2.5: W-MAC limited by the short range of WuR

2.4 Challenges

Unfortunately, the main drawback of the WuR technology is that the sensitivity is lower than that of the main radio, even with low data rates [1]. This means that the range is shorter, ranging from 2 to 20 m according to [19], which translates into a very dense network. This can result in situations in which two nodes can communicate together with the main radio, but not with WuR. In such a situation, a node cannot wake up its destination and therefore communicate with that node while they are in range regarding the main radio. Using the simplest approach, W-MAC [17], the resulting range of the system is upper bounded by the WuR range. That means that a source may need to relay a data packet over intermediate hops to reach its final destination, as depicted in Fig. 2.5, even when source and destination could communicate at only 1 hop on the main radio, according to its range.

To overcome this problem, one way is to increase the output power of the transmitter to compensate for the low sensitivity. However, this results in huge power consumption that cancels the benefits of the WuR [20]. Another option is to deal with the range mismatch by routing the WuS through intermediate nodes to wake up the destination and then transmit the data packet directly in a single hop on the main radio [21], [22].

Let's assume we have 3 nodes as depicted in Fig 2.6. Considering the range mismatch, node 1 is connected to node 2 through the WuR but it is not connected to node 3. On the contrary, node 1 is connected to all the nodes through the main radio. In other words, there are two parallel topologies: the one defined by the WuR links (very short range) and the other one defined by the main radio links (medium range). In that example, if node 3 wants to send a message to node 1, it cannot do it using WuR because there is no direct link on the WuR channel between them to send a wake-up signal and wake it up. However, if node 2 serves as a relay for the WuS, then node 3 can communicate data with node 1, as depicted in Fig 2.7. Node 3 first sends a wake-up signal to node 2 (step 1). Upon reception, node 2 forwards this wake-up signal to node 1 and keeps its main radio in sleep (step 2). Then, node 1 wakes up its main radio and starts listening to the channel waiting for the data frame. Finally, node 3 transmits the data (step 3). From now on we define the amount of time between step 1 and step 3 as the *sync delay* because node 3 is

[1] -55 dBm for the WuR against -90 dBm to -110 dBm for the main radio.

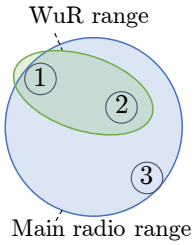


Figure 2.6: Wake-up radio and main radio ranges

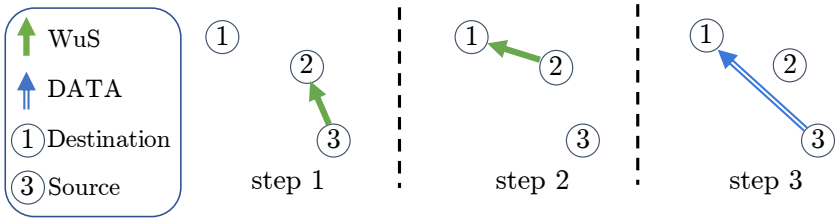


Figure 2.7: Example of ideal data communication

waiting for node 1 to wake up. Node 1 may reply with an acknowledgment frame on the main radio (not shown in the figure).

This means that the nodes need to find routes on the WuR topology to communicate with some of their main radio neighbors. This is what we understand by *wake-up signal routing* or *the routing layer in wake-up radios*. For this reason, we need to choose a strategy to route the wake-up signal on the WuR topology. It is a hard challenge to find the shortest path in such scenarios because we have to take into account both topologies. Then, we encounter the problems of calculating the sync delay and the fact that the size of the wake-up signal has to be limited because of the low data rate.

2.5 Related work

A complete survey of the WuR technology was published in [3]. We conclude from that article that most of the work done so far in WuR is mainly focused on the hardware side and not too much attention has been paid to the networking counterpart. Even the protocols surveyed in that paper are in general very specific to a particular prototype or application [23], [24].

An interesting comparative analysis between preamble sampled MAC protocols and WuR receivers in WSN has been proposed in [20]. There, it is shown that in practice, the range mismatch requires a denser deployment of the WSN, or the need to transmit the WuS at a high output power. However, there is no further analysis of the details of the communication protocol when the node density is increased. This is likely to increase the competition in the WuR medium, leading to more collisions. As we are going to see in Section 3.4, the reliability of the WuR medium significantly impacts the performance of WuR. Nevertheless, the main drawback of that article is that the analysis considers outdated hardware prototype specifications, and it does

not consider relaying the WuS to deal with the range mismatch.

Simulations in OMNeT++ [25] have been done in [26] to prove that WuR constantly allows for substantial energy savings, higher packet delivery ratio, lower latency, and less complicated software implementations. Even though the authors provide several variations of the network type and details on how to reproduce the analysis, the comparison considers neither the relay of the WuS to reach a destination nor acknowledging WuS packets. Relaying the wake-up signal improves energy consumption because it reduces the number of transmissions on the main radio, as we show in this thesis.

T-ROME [27] is one of the few solutions that propose to use intermediate nodes to retransmit the WuS between the source and the destination. Then, each node that receives a copy of the wake-up signal sends back a Routing Acknowledge packet containing some metrics such as its link quality identifier. This information allows the source to chose the best relay node to deliver the data packet towards the sink. Besides, it allows sending several main data packets in a row once a communication link is established. However, the authors do not analyze how large can the network be or whether the use of acknowledgments on WuR increases performance and reliability.

W-MAC [17] was introduced in Section 2.3. This solution is the only one in the literature that has been proved to work along with the Routing Protocol for Low Power and Lossy Networks (RPL) [12]. RPL is the Internet Engineering Task Force standard for multi-hop routing in WSN. This protocol builds a destination-oriented directed acyclic graph based on distance vectors, and each node selects a preferred parent when joining the network. While W-MAC achieves a great power efficiency, latency, and reliability compared to the duty-cycled approach, it does not fix the main problems of RPL, such as inefficient parent selection and slow recovery time after a preferred parent dies. These challenges are further explored in Section 5.1.

G-WHARP was presented in [28] as a data forwarding solution that combines the benefits of energy harvesting and WuR with semantic addressing to optimize energy efficiency. Forwarders of the data packet are selected according to a Markov Decision Process that learns to minimize the power consumption and latency, as well as maximize the packet delivery ratio. Unfortunately, they do not address the range mismatch between the main radio and WuR. As a result, the communication is upper bounded by the short range of WuR. That means that only the nodes that are in the WuR range of a source can be selected as data forwarders. Moreover, they only compare to a proactive routing protocol, CTP-WuR, that addresses the range mismatch challenge.

Ghose et al. presented BoWuR in [29], CCA-WuR, CSMA-WuR and ADP-WuR in [30]. In those works, they demonstrate the importance of using CCA and backoffs to improve WuR performance under high traffic loads. However, they do not explore the impact it has when relaying the WuS to address the range mismatch or when calculating the sync delay.

In [9], the authors proposed a proactive approach, called *FAWR*, using WuR to route the WuS with a policy power manager for energy harvesting systems. It is presented in the context of the receiver-initiated types of protocols. There, a centralized base station coordinates the data transmissions of each end-device and its main advantage is that it reduces the chances of collisions. All the nodes in

the network are in the main radio range, but since the WuR range is around 10 times shorter, the WuS has to be relayed through intermediate nodes. The network performance is evaluated through simulations in OMNeT++ and compared to five traditional duty-cycled MAC protocols and one WuR-based protocol. It is also validated with an experiment of 3 real nodes. The protocol presented is an efficient solution that avoids collisions by coordinating the packet transmissions from the base station. The authors state that synchronous protocols are not well suited for energy-harvesting WSN because of the synchronization overhead that wastes energy. However, the proactive approach that they present also requires control overhead and maintenance to coordinate the data transmissions. Furthermore, it does not address the challenge to calculate the sync delay.

Another proactive approach for routing the WuS is presented in [31] in the context of CTP-WuR [32], a protocol that addresses the range mismatch problem in WuR networks. In this work, they propose two phases: Discovery and Exploration. The algorithm presented requires all nodes to exchange link-state information across the whole network and compute a minimum-cost tree. Their results prove that multi-hop relaying considerably improves energy efficiency, against not relaying the WuS. However, this protocol was only compared to traditional duty-cycled MAC protocols. In particular, it was not compared to any reactive WuR-based protocol. Also, the authors do not mention how to compute the sync delay when relaying the WuS.

OPWUM [33] is a reactive opportunistic algorithm to select the next-hop at the MAC layer. It uses an RTS/CTS/ATS mechanism to avoid collisions and to retrieve a given metric from the receivers to select the best one. However, this algorithm only works for a single hop of the WuS, that is, it does not address the range mismatch in WuR. It does not provide a way to find the following hops of the WuS to traverse a route from a main source to the final destination.

2.6 Conclusions

In conclusion, a lot of work has been done comparing WuR with duty-cycled MAC protocols, but we identified some gaps that have not been investigated yet, with special emphasis on the range mismatch and the computation of the sync delay challenges. We expect that WuR-based solutions can achieve pure-asynchronous communications that are easy to implement, deploy, and maintain while keeping a good quality of communications. These benefits are most noticeable at the MAC layer, but its usage has a significant impact on the routing layer.

Furthermore, the range mismatch is the first challenge if we want to use the main radio at its full power. There is a need for a control mechanism to compute a layer-2 route to reach the destination of a frame. This also introduces the sync delay problem that has not been addressed in the literature so far. Then, routing over a two-tier architecture or a multi-path is a complex task as WuR and main radio topologies partially overlap. In the past, some protocols have been proposed to route the wake-up signal in WuR networks in proactive and reactive ways. Nevertheless, none of them propose a comparative study between both approaches addressing the range mismatch problem. Similarly, the following question remains open when designing the network stack with WuR: is proactive or reactive the best approach

for wake-up signal routing?

In this thesis, we analyze the benefits and drawbacks of using WuR in multi-hop WSN. In the following chapter, we start by proposing two MAC protocols to use WuR and investigating the limits of the WuR technology at the MAC layer. Then, our contributions to the routing layer are presented in Part II.

Part I

First contribution: Investigating the limits of WuR at the physical and MAC layer

Chapter 3

Is the latency-energy tradeoff over?

Contents

3.1 Introduction

The main drawback of the ultra-low power consumption of WuR is that the sensitivity is dramatically decreased compared to existing technologies for wireless communications, such as 802.15.4-compliant radios. If the sender transmits at a higher output power, the range may be increased to match that of the main radio. The problem in that case is that the benefits of the ultra-low power consumption of WuR are cancelled by the high power consumption on the sender. Furthermore, in some cases those high levels of transmission output power may exceed regulatory requirements [20].

However, if the sender transmits at the nominal power, then the range of the WuR is shorter than that of the main radio. Consequently, the network space density must be higher, for the nodes to be connected with such a shorter range. As a result,

waking up a destination may require to relay the WuS through intermediate nodes, potentially increasing the end-to-end latency.

In addition, the hardware design of the WuR and the modulation used makes us wonder how robust it is against interferences or collisions. This is especially challenging because of the network density and the long time over the air, due to the low data rate, that increases the contention.

Remarkable comparative analysis with duty-cycled MAC protocols have been done in the literature, including [20] and [26]. However, to the best of our knowledge, there has not been any study on the challenges cited above.

Contribution

- We propose two protocols that use the WuR at the MAC layer: WuSone-way and WuSACK.
- We investigate the benefits, drawbacks and tradeoffs of using WuR in multi-hop WSN in terms of the packet delivery ratio, latency and power consumption.
- Our results show that there is a threshold in the size of the network for WuR to perform efficiently. Increasing the network size beyond this threshold significantly degrades the WuR performance, making a traditional duty-cycled MAC protocol a better choice for such configuration.
- We also show that acknowledging the WuS is problematic in the presence of collisions, because it decreases seriously the reliability of the network.

Publication: S. L. Sampayo, J. Montavont, F. Prégaldiny and T. Noel. Is Wake-Up Radio the Ultimate Solution to the Latency-Energy Tradeoff in Multi-hop Wireless Sensor Networks?, in the 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Limassol, Cyprus, October 2018, rank B.

3.2 Wake-up radio MAC protocols

3.2.1 Naming convention

We will use the following terms throughout this chapter:

- Scenario: a fixed number of nodes running a specific protocol in a fixed medium model
- WuS Request (WuS REQ): WuS sent from the source to the destination

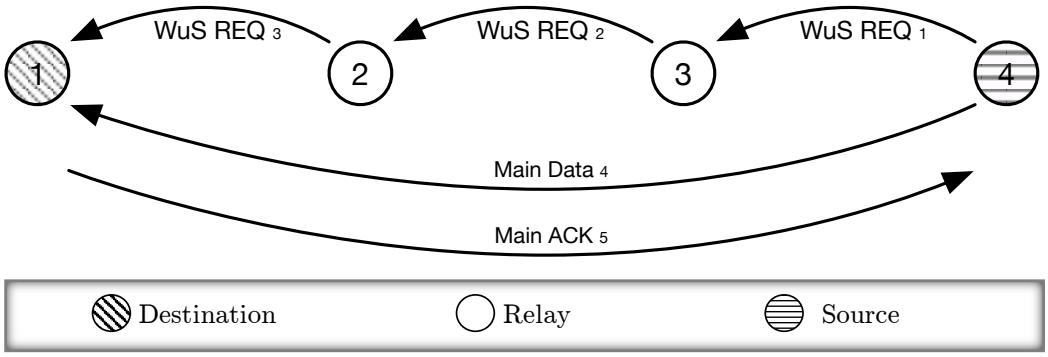


Figure 3.1: WuSone-way protocol

- WuS Acknowledgment (WuS ACK): WuS sent from the destination to the source
- WuS path: The path of nodes that the WuS has to travel to reach its destination
- WuS sequence: The sequence of retransmissions of WuS that needs to be made to travel the path
- Main data: Main data frames sent over the main radio
- Main sender: The source node of the communication, the one who desires to send a data frame
- Main destination: The destination node of the communication, the one who should receive the data frame. It is also the final destination of the WuS path

3.2.2 WuSone-way protocol

In this chapter, we begin with a well-known line-shaped network where we have one source, one destination, and a variable number of nodes between them in a row, only acting as WuR relays. Fig. 3.1 shows an example with two relay nodes. Main sender (node 4) and destination (node 1) are placed at the opposite borders of the line-shaped network. The main radio range is large enough so that both are connected through a single hop. However, the WuR range is small enough so that only adjacent nodes are connected. That means that for 4 to wake up 1 it needs to send a WuS to 3, and 3 needs to retransmit it to 2, and finally, 2 needs to retransmit it to 1. This operation is illustrated in Fig. 3.1. Such a situation should be very common in real life deployments due to the range mismatch between the main radio and WuR. However, analyzing the impact of relaying WuS has never been investigated before.

A short time after the WuS was sent, the main sender transmits the data packet over the main radio. We have defined previously this short amount of time as the *sync delay* because node 4 is waiting for node 1 to wake up. After the data is received by the destination, an acknowledgment is sent back to the sender. Fig. 3.2 illustrates this single mode of operation. Notice that a longer WuS path makes the destination wake up later. So if the sync delay is not long enough the data will be

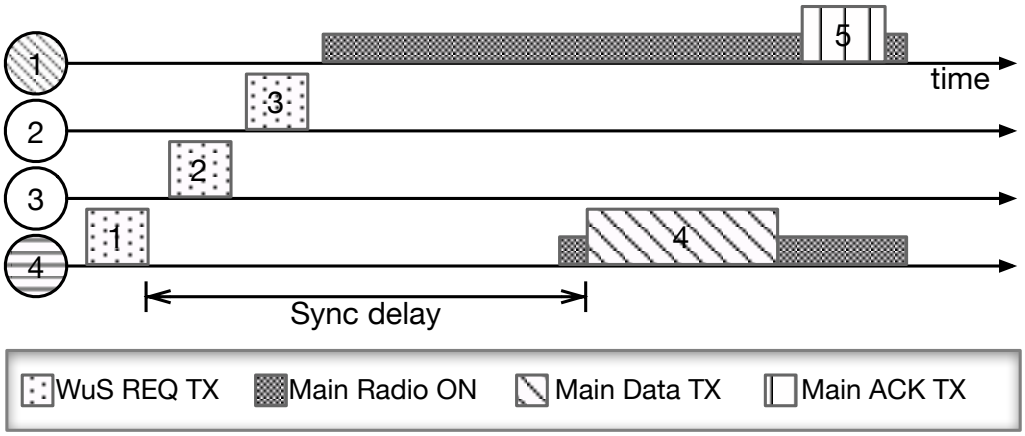


Figure 3.2: WuSone-way protocol timeline

transmitted when the destination is still sleeping, requiring a retransmission of the frame.

The sync delay can be fixed as a constant value during compile-time or it can be calculated dynamically in run-time. Another option is to let the destination transmit an acknowledgment on the wake-up radio upon reception of the WuS, so that the source is notified when the destination is ready to receive the data packet.

3.2.3 WuSACK protocol

The usage of WuS Acknowledgments (WuS ACKs) is not clearly pointed out in the literature. It is in some cases used as a response of a potential next hop in reactive protocols [33], but there is no indication on an end-to-end acknowledgment of the WuS through several hops. This way it would be possible to avoid the retransmissions problem related to a long WuS path and late destination wake up. Furthermore, it avoids the use of the main radio in vain if the WuS does not reach the destination. For this, we introduce WuS ACK frames. The WuS ACK is sent by the main destination when it receives a WuS REQ and wakes up. This type of packet is relayed in the same fashion as the WuS REQ but in the opposite direction, towards the main sender. Upon reception of the WuS ACK, the sender is assured that the final destination is ready to receive the pending packet. This means that, in our example, firstly there is a WuS sequence from node 4 to node 1, secondly a WuS ACK sequence from node 1 to node 4, and only after both are successful, the main data and data ACK are exchanged between main node 4 and node 1. This scenario is illustrated in Fig. 3.3.

3.3 Simulation framework

There are lots of simulators and simulation frameworks for WSN in general [25] [34], but not so much has been done in the area of multiple radios. Several authors have been working with GreenCastalia [35], an extension of OMNet++ [25], that allows to easily model and simulate networks of embedded devices with energy-harvesting

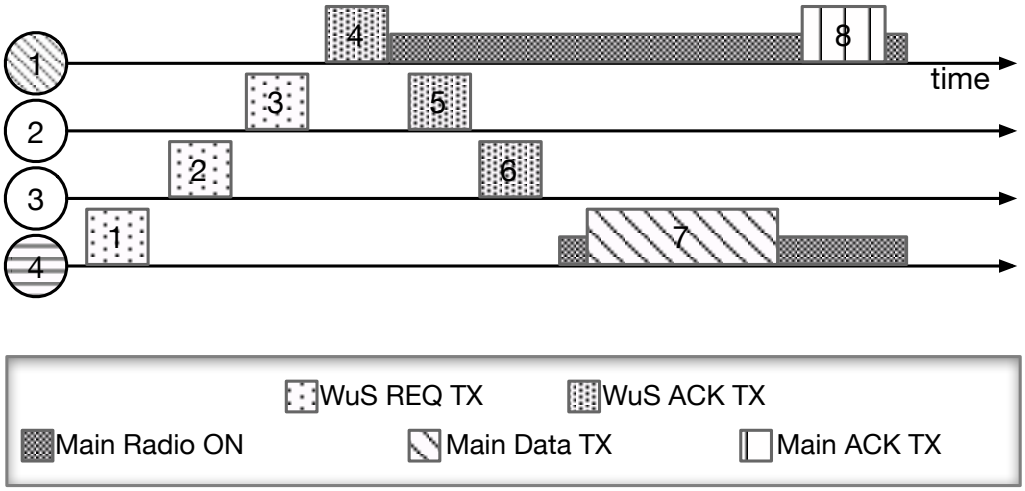


Figure 3.3: WuSACK protocol timeline

capabilities. However, this software cannot reproduce exactly the firmware that runs on the real devices and it does not consider the entire stack. In this work, we used WaCo [17], a COOJA extension that models a WuR prototype proposed in [16], and implements a WuR MAC protocol called W-MAC. COOJA is a software that simulates a network of ContikiOS nodes, allowing us to easily port the code for real devices. WaCo's performance was validated against experiments and it emulates a real prototype [17]. Although this software provides the basic elements to handle the WuR, there is no support for relaying WuS packets or changing the RX success probability of the WuR medium. These features have been added to WaCo and the W-MAC layer in ContikiOS in this contribution. Note that WaCo supports only SkyMote nodes [36], which uses a CC2420 transceiver for wireless communication.

3.3.1 Simulation setup

Table 3.1 summarizes the most important simulation parameters. For the rest, we use the default values of Contiki parameters. We analyze the performance of a line-shaped network changing the total number of nodes: one main sender, one destination and some nodes in the middle, ranging from 0 to 8, whose only mission is to retransmit the WuS. We also vary the collision probability in the medium model separately for each radio type. In all cases, we perform 6 simulation runs for each particular scenario and we cut off the first and last 10 s of the duration of each simulation.

We implement 3 main MAC protocols for the analysis:

- **ContikiMAC** We use ContikiMAC [38] with the default parameters and turn on the phase optimization mechanism. For scenarios with more than 2 nodes, the nodes that are placed between the sender and the destination do not participate actively in the communication (but may suffer from idle listening and overhearing).

- **WuSone-way** In this protocol, only the WuS REQ is sent to wake-up the

Table 3.1: Simulation parameters

Parameter	**Value**
Simulation duration	120 s
Repetitions of each simulation	6
ContikiMAC channel check period	125 ms
MAC layer	CSMA (Contiki version)
Max CSMA retransmissions	7
Network layer	RIME [37]
Packet rate	1 packet/s
WuS packet length	16 bits
WuS data rate	100 kbps
Main data packet length	43 bytes
Main data ACK packet length	5 bytes
Main radio data rate	250 kbps
Main Node	Sky mote [36]
WuR HW prototype	[16]
WuR Supply Voltage	1.8 V
WuR TX current	8 mA
WuR RX current	2 mA
WuR idle listening power consumption	1.944 μW
Main Radio medium model	UDGMConstantLoss
WuR Radio medium model	UDGMConstantLoss
Main radio RX success ratio	100%, 80%
WuR RX success ratio	100%, 90%, 80%
WuR Sync delay	1.8ms, 3.1 ms, 6.45 ms

main destination. The main data is sent after the sync delay described in Section 3.2. We implement 3 versions of this protocol: short sync delay (data transmission as soon as possible), medium sync delay and long sync delay. In the long case, the delay is long enough so that the WuS REQ can hop up to 9 times (which is the maximum distance in the line-shaped network of our setup) and arrive successfully at the main destination before the main data is sent. The WuS REQ contains the node number of the next relay on the WuS path towards the destination.

- **WuSACK** Here when the main destination receives a WuS REQ, it transmits a WuS ACK back to the main sender. The sender will not transmit the main data until it receives the WuS ACK back. The WuS ACK contains the node number of the next relay on the WuS path back towards the source.

For all of them, we use the Contiki implementation of CSMA on top of them without backoff exponential increments. RIME is used as the network layer for the main node in all scenarios because it is a lightweight protocol that let us focus on the MAC layer.

The currents and voltage values for the Sky platform are the nominal values obtained from the "Typical Operating Conditions" table of the datasheet [36]. On the other hand, the values for the WuR are borrowed from the experimental validation in [17]: 1.8 V of power supply, 8 mA current when transmitting the WuS, around 2 mA when actively receiving a WuS, and 1.944 μW power consumption when idle listening.

To measure the power consumption of the main radio and main MCU, we use a combination of Powertrace [39] and PowerTracker. For the power consumption of the WuR in TX and RX modes, we use the WurPowerTracker plugin provided in [17]. The idle listening power consumption of this secondary radio is not measured directly as it is always-on, so it is a constant added manually to our post-processing.

3.4 Results

The results presented in this section are an average of the overall data collected on the set of simulations. The 95% confidence interval indicates the reliability of our measurements.

3.4.1 Packet delivery ratio

We see in Fig. 3.4 that the WuS sequence length strongly affects the PDR. This is because it is more difficult to get several packets in a row successfully than just a few of them. The PDR decreases with the increase of the network size, but also with the addition of WuS ACKs, which doubles the amount of WuS packets for each scenario reducing significantly the reliability of the communication. In consequence, the PDR is tightly controlled by the RX success probability of the WuR medium. In addition, we present the resulting PDR when the probability of collisions for the WuR medium is decreased to 10%, while the corresponding probability for the MR medium is kept at 20%, in Fig. 3.5. As we can see the PDR significantly increases for

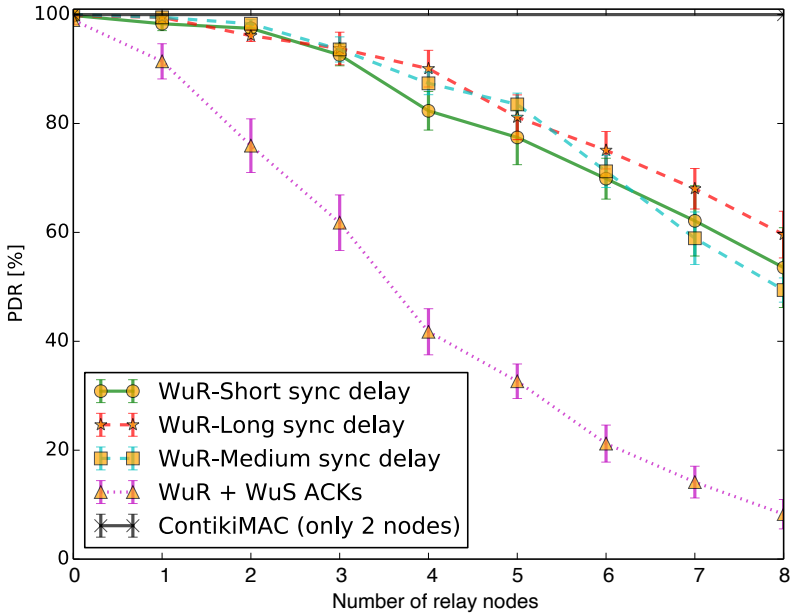


Figure 3.4: Packet Delivery Ratio. 20% probability of having collisions.

all WuR solutions, emphasizes that the reliability of the WuR medium is of crucial importance.

The PDR for the ideal medium is not shown because it is always 100%. In fact, even if the MR medium presents collisions, but the WuR is ideal, then the PDR is still 100% thanks to the retransmissions allowed by CSMA. This reinforces the conclusion that the PDR is mandated by the number of collisions in the WuR medium.

3.4.2 Latency

The latency behavior is led by the number of main data retransmissions. This is why the shape of the lines for each protocol in Fig. 3.6 is similar to that of the power consumption. So again, WuR-Short sync delay is the best one for small network sizes, WuR-Medium delay is so for medium networks and WuR-Large delay for large networks.

In contrast, when there is a chance of collisions in the medium, it is always better to provide the minimum amount of intentional delay to the protocol, as shown in Fig. 3.7. This is because each retransmission adds this delay to the total latency, so if there are going to be several retransmissions, it is better to add the minimum delay possible for each one. It is noticeable that WuR + WuS ACKs provides the worst performance in terms of latency because is the one that requires more number of retransmissions for each successful communication.

ContikiMAC latency is not plotted because it is very high and the figure is easier to read without it. Its value is around 100 ms for the ideal medium and 120 ms

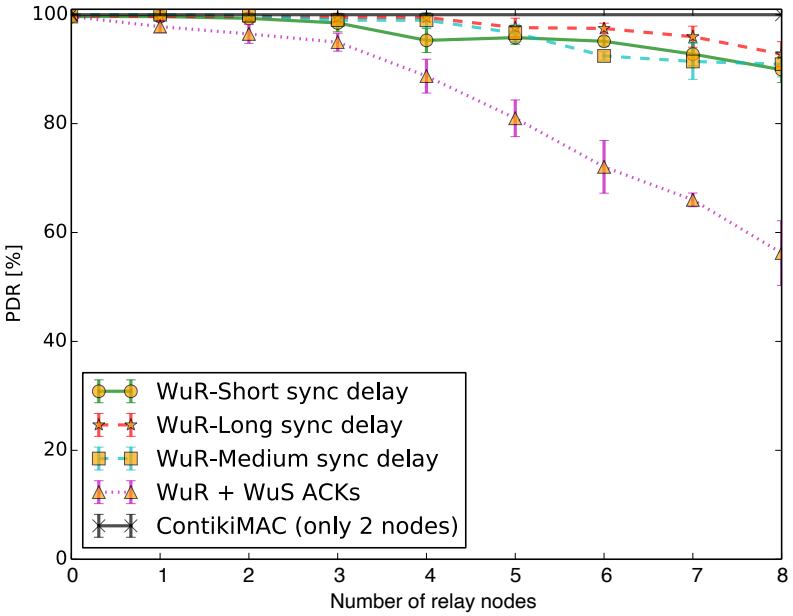


Figure 3.5: Packet Delivery Ratio. 20% probability of having collisions in the main radio and 10% in the WuR

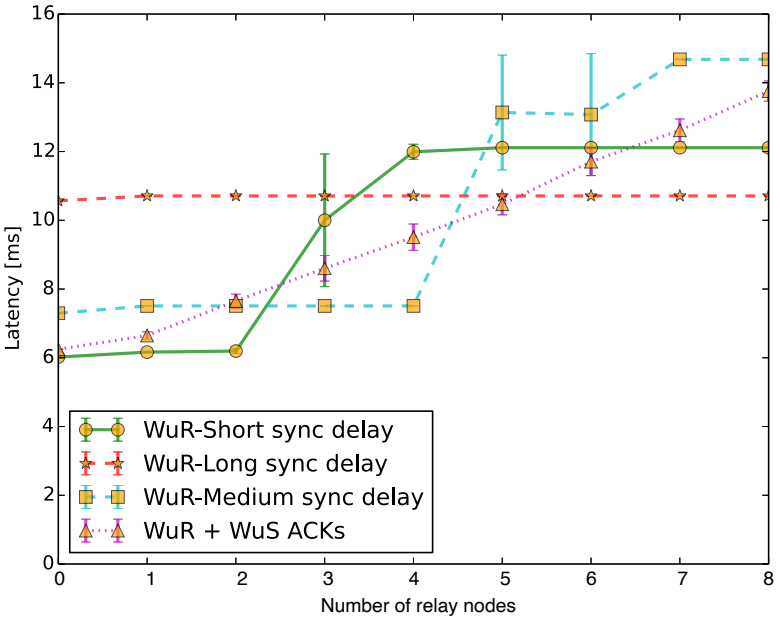


Figure 3.6: End-to-end latency. Ideal radio medium (no collisions).

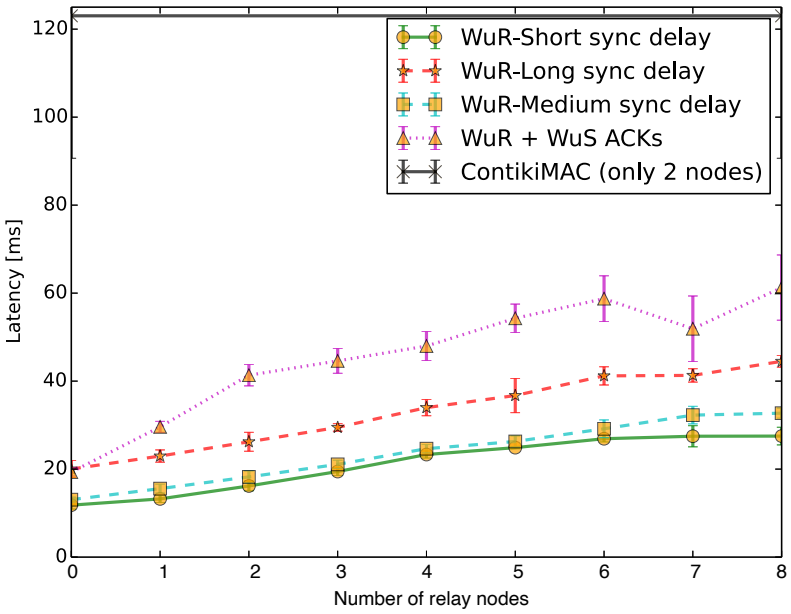


Figure 3.7: End-to-end latency. 20% probability of having collisions.

for the collision scenario (20% for both mediums). This is a reasonable number, considering that we used the default channel check rate of ContikiMAC which is 8 Hz. This corresponds to a period of 125 ms.

3.4.3 Power consumption

3.4.3.1 Ideal mediums

The power consumption results in the ideal medium are shown in Fig. 3.8, where there are no interferences or collisions. The network radio mean power consumption only takes into account transmission, reception and idle listening modes for power consumption calculation, i.e., it does not consider the CPU and Low Power Mode (LPM). The values shown in the figure are the total sum of all nodes in each scenario.

The only scenario that is plotted for ContikiMAC is the 2-nodes network (0 relay nodes), which is extrapolated horizontally, because all the other scenarios running this protocol always consume more energy than its WuR counterpart. This is due to the large amount of idle listening and overhearing that is spent on the main radio in duty-cycled protocols like ContikiMAC. This is avoided with the WuR because the idle listening is done with the ultra-low power radio. In the WuR protocols, there is no overhearing of the main radio because if the node is not the main destination, then its main radio stays off. In that scenario (0 relay nodes), WuR-based protocols can reduce the overall power consumption by a factor of 2 compared to ContikiMAC.

Among the WuR protocols, we can divide the scenarios into 3 types depending on the network size (number of nodes): small, medium and large. In small networks,

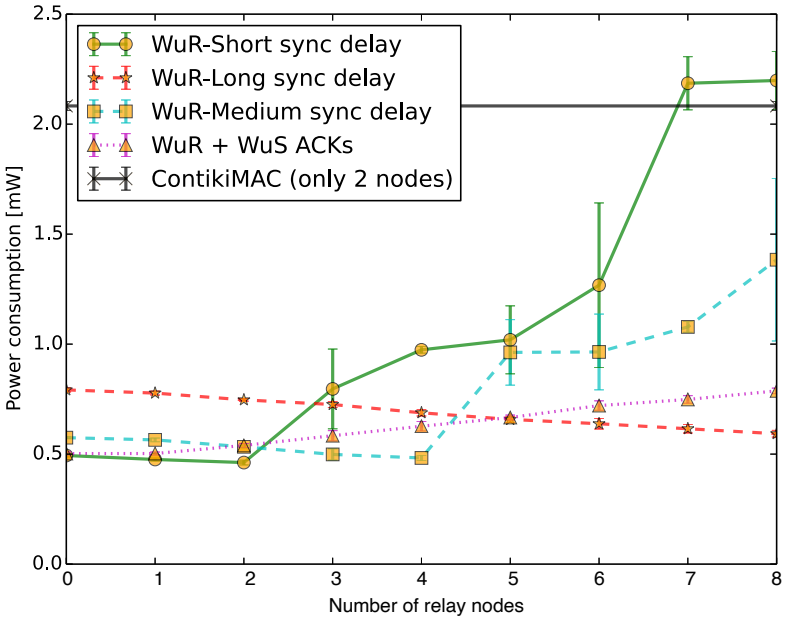


Figure 3.8: Network radio mean power consumption. Ideal radio medium (no collisions).

we see that WuR-Short sync delay is the protocol that performs the best in terms of power consumption, while the power consumption of protocols with intentional sync delay increases. This is because in a small network the main destination is near, so it receives the WuS and wakes up quickly. Then, the sync delay makes the destination wait in idle listening, wasting energy. As the number of relay nodes goes up, the WuS path length increases, but the sync delay at the transmitter stays the same. Hence, the idle listening wastage is reduced as the network size increases. This explains why the power consumption of WuR-Long delay decreases whereas the network size increases. WuR + WuS ACKs solves this problem because the sender gets to know in real time when the main destination is awake so it can send the main data.

WuR-Short sync delay deserves special attention at a medium network size of 3 relay nodes, where we see a sudden jump in the power consumption. As we increase the network size, there is a point in which the WuS path is so long that by the time the sender transmits the main data, the main destination has not received the WuS yet, so it is not awake and therefore is not ready to receive it. Consequently, the sender needs to retransmit the main data, wasting energy in useless early transmissions. When that happens, the transmitter starts over from the very beginning, sending the WuS and later on retransmitting the main data. After some fixed timeout following the reception of a WuS, the destination goes back to sleep.

For WuR-Medium sync delay at that network size, this problem is avoided by the addition of the sync delay, which makes it the best protocol in terms of power

consumption for medium network size. As a matter of fact, we can see that if we keep increasing the number of relay nodes, there is a new jump in the power consumption at some point (for 5 relay nodes). On the other hand, the long sync delay is long enough so that the problem does not arise even in a 8-relay-nodes network. Finally, in the WuS ACKs counterpart, this problem is not present for the same reasons as in smaller networks.

In large networks, there is another special point in the WuR-Short sync delay solution, at the 7-relay-nodes scenario, where we find a new jump in the power consumption. In this case, the problem is that the WuS path is quite long. The second sequence of WuS REQ arrives at the main destination after the main data and ACK have been exchanged, so the destination wakes up again and waits for a new data packet that will never arrive because the sender has already sent it successfully. This increases the amount of idle listening on the main destination resulting in a higher power consumption.

In large networks, WuR-Long sync delay outperforms all other protocols because its long sync delay prevents the sender from retransmitting the main data. Finally, WuR + WuS ACKs consumes more energy than WuR-Long sync delay because it needs more WuR packets to be sent (for WuS ACKs), increasing a little bit the power consumption in the WuR, while the power consumption of the main radio remains the same. However, WuR + WuS ACKs seems to be the best compromise regardless of the number of the number of relay nodes in the network.

3.4.3.2 Collisions

We run the same scenarios by adding a probability of collisions of 20% in each radio medium. As a result, each transmission (either on the main radio or WuR) have a probability of 80% to be successfully received by the destination. The results are shown in Fig. 3.9. The overall power consumption is higher because of the retransmissions of the main data, but the behavior, in general, stays the same.

There is one particular change in behavior in WuS + WuS ACKs, where we can see that for small networks, there is a very high power consumption that increases proportionally with the network size. This is because most of the times the WuS sequence is successful, so the main destination wakes up. However, later on, some WuS ACK packet fails so the main sender does not transmit the main data. Meanwhile, the destination keeps listening on the main radio in vain. In consequence, the energy consumption increases due to idle listening on the main radio at the main destination. This is also reflected in Fig. 3.10, where we see clearly that the total power consumption is commanded mainly by the destination node. In conclusion, we can think of an hybrid approach combining WuR + WuS ACKs and WuR-Long sync delay. If no WuS ACK has been received after the long sync delay expires, then the source could send the data anyway.

Whereas when the network is larger, there are several nodes in the middle and it becomes very hard to get all the WuS sequence successfully to wake-up the final destination, so there is less listening wastage. In addition, it is even more difficult to get the full WuS + WuS ACK sequence successfully to get to send a main data packet. So in the end, there is less listening on the destination and less number of main data packets transmitted by the sender, which explains why the power

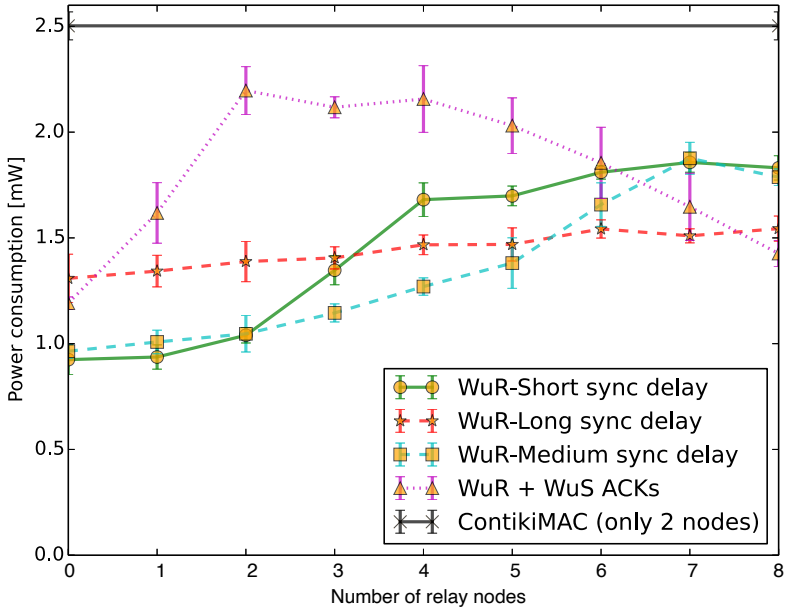


Figure 3.9: Total radio mean power consumption. 20% probability of having collisions.

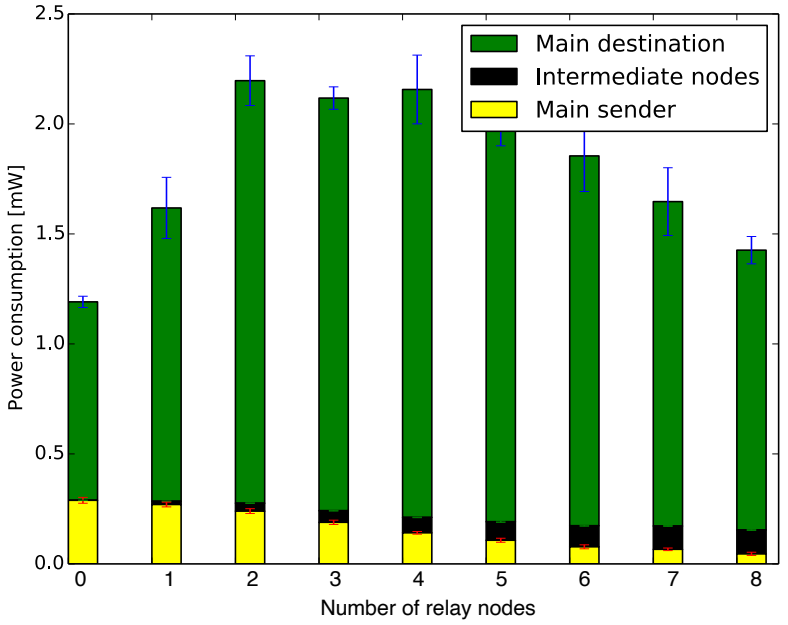


Figure 3.10: Power consumption per type of node. Protocol WuR + WuS ACKs. 20% probability of having collisions.

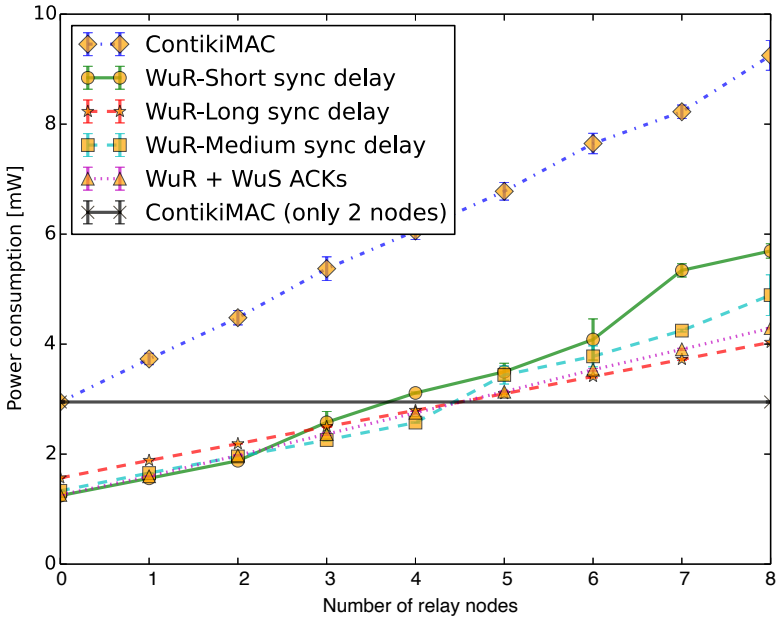


Figure 3.11: Network mean power consumption. Ideal radio medium (no collisions).

consumption of the main sender decreases while the network size increases. This behavior is because main radio packets are the ones that impact more on the overall power consumption while WuR packets retransmissions are insignificant.

3.4.3.3 2-nodes ContikiMAC comparison

We turn our attention now to figure 3.11, where we do consider the CPU and LPM to compare the total power consumption of the network. We can see in this figure that there is a threshold in the network size that determines whether the WuR outperforms ContikiMAC or not, when we compare the WuR protocols with the initial ContikiMAC scenario with only 2 nodes and 0 relays. Such scenario corresponds to a power consumption of approximately 3 mW. The horizontal black line at that value let us compare such scenario with the WuR protocols easily. This threshold turns out to be in a network size of between 5 and 7 nodes, corresponding to 3 to 5 relay nodes. So for example, this means that having 2 nodes communicating with ContikiMAC is better in terms of power consumption than having 7 nodes communicating with a WuR protocol (where 5 of them are relay nodes). In contrast, 2 nodes with ContikiMAC is worse than 4 nodes with a WuR protocol (where 2 of them are relays). This threshold barely changes for most protocols when we increase the probability of collisions. WuR + WuS ACKs is the only one that presents a large variation of this threshold, because of the high power consumption for small networks that we described previously.

On the other hand, the blue line is the result of using ContikiMAC for each network size (with more than 2 nodes). In that case, the intermediate nodes do not

relay anything because WuR is not used and source and destination communicate directly. However, they consume power because of the fact of being turned on. If WuR-based protocols are compared to those cases, we can see that WuR outperforms ContikiMAC for all scenarios.

3.4.4 Simulation conclusions

In the light of the obtained results, some conclusions can be drawn. First, WuR is not always the final solution to the latency-energy tradeoff in multi-hop WSN. As expected, WuR achieves better power consumption and latency than traditional duty-cycled protocols, in particular, ContikiMAC, considering an ideal medium with the destination directly in the WuR range of the sender. However, due to the range mismatch between MR and WuR, WuS packets would be relayed by intermediate nodes to reach the final destination. In that case, WuR still achieves better performance if the relaying nodes involved in the communications are no more than 5, as shown in Fig. 3.11. For large networks (for 6 or more relaying nodes), the latency obtained with WuR is still lower than for ContikiMAC, but the overall power consumption is higher. The reason for that is that the contribution of the WuR to the total network power consumption becomes more significant as the number of nodes relaying the WuS goes up. Collisions on the WuR medium also have a very significant impact on overall performance, especially on the achieved PDR. It is of crucial importance that WuS packets reach the destination or WuR presents severe underachievements. We are considering mechanisms such as an exponential backoff on WuS packets to mitigate this. Finally, we advocate to not use acknowledgments for WuS packet. Not only the power consumption is slightly increased, but the PDR is significantly reduced together with an increase in the latency. Obviously, WuR-Short, WuR-Medium and WuR-Large sync delays respectively match small, medium and large networks. We will investigate how we can dynamically adapt this delay regarding the network size, or limit the number of relay nodes to 3 (because it outperforms duty-cycled MAC protocols) and use WuR-Medium sync delay.

3.5 Conclusions

In this chapter, we investigated how Wake-Up radio can increase the network performance of multi-hop WSN. For this, we analyzed the power consumption, latency, and PDR of a traditional duty-cycled MAC protocol (ContikiMAC) and 4 variations of the WuR protocol. In Section 3.4, we find that WuR protocols always outperform traditional duty-cycled MAC protocols for the same number of nodes. Although if we compare a scenario with only 2 nodes using duty-cycle, then there is a threshold between 3 and 5 for the number of nodes that we can have in the middle relaying the WuS for it to be outperforming.

We show in Section 3.4 that the presence of collisions significantly affects the network performance, wasting energy and degrading the PDR and the latency. In fact, WuS ACKs is only worth it if the WuR medium is free of collisions. Using exponential backoff in CSMA can mitigate this issue on the WuR at the cost of increasing the latency.

A protocol that adapts dynamically the sync delay, according to the network size,

can achieve the minimum power consumption. The drawback is that the latency would be worst in real scenarios where we have collisions. Our results show that, in the collisions case, it is better to have the minimum sync delay possible no matter the network size. So here we show that there is still a tradeoff between power consumption and latency in WuR.

Medium delay protocol appears to be a reasonable compromise between latency and power consumption for collision scenarios.

As for the power consumption, we show that it is not the retransmission of WuR packets what impacts more, but the retransmission and idle listening on the main radio. In conclusion, it does not seem to be necessary to duty-cycle the WuR, as is suggested in some works [3].

In the next chapter, we are going to extend our work with experiments on hardware prototypes to understand better the nature of the WuR links and model them accordingly in our simulation framework.

Chapter 4

Experiments in a noisy environment

Contents

4.1 Introduction

In this chapter, we present an experimental platform using the WuR prototype described in the introduction, and analyze its behavior when it is subject to radio interference. We remind that this prototype consumes around 1.2 μW and its sensitivity of -55 dBm translates into a range of around 20 m, according to [40], at 868 MHz. It is important to consolidate our understanding of the bottom layers of the stack before addressing the challenges that appear at the routing layer. The lessons that we learned in this chapter are going to be applied throughout the rest of the thesis. In particular, they help tuning the parameters of the simulations in other chapters according to the real-world behavior.

Contribution

- We perform a study on the behavior of a WuR prototype when it is subject to a real-world noisy environment. We analyze how interference can be wrongly considered as valid packets and how to deal with them.
- We show the importance of the utilization of Clear Channel Assessment (CCA) capabilities to reduce transmission errors, resulting in a higher Packet Delivery Ratio (PDR).
- We extract some key physical values of the prototype that can serve in modeling WuR communications.
- We provide a method to estimate the overall current consumption of an application deployed over a WuR-based network.

Publication: S. L. Sampayo, J. Montavont and T. Noel. A Performance Study of the Behavior of the Wake-Up Radio in Real-World Noisy Environments, in AWAKE workshop of the 2020 International Conference on Embedded Wireless Systems and Networks on Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks (EWSN), Lyon, France, February 2020.

4.1.1 Background

In [40], the authors integrated a WuR prototype [16] into a wireless mote and performed transmission range measurements between two nodes. They showed that the maximum transmission distance is around 21-24 m while the packet delivery ratio remains above 95%. They also categorized wake-up signals as *Positives* - wake-up signals that should be considered as valid by the receiver, and *Negatives* - wake-up signals that should be discarded by the receiver. *Positives* could be splitted in *True Positives* - wake-up signals that were effectively sent by a valid source, and *False Positives* - wake-up signals that were not transmitted by a valid source. Generally, noise or external interference are the source of *False Positives*. *Negatives* includes *False Negatives* - wake-up signals that were transmitted by a valid source but were corrupted somehow, and *True Negatives* - wake-up signals that were not transmitted by a valid source. They showed that *False Negatives* remain low (below 1%) while *False Positives* are negligible (stand for 0.02%). However, the *True Negatives* in [40] are generated artificially by manually corrupting the original wake-up signal. This leads to a high percentage of *True Negatives* in the results, which only means that the receiver decodes correctly those artificially-generated wake-up signals. In our study, we analyze the whole set of *Negatives* that are completely generated by noise or external interference, approaching more realistic scenarios. We qualified and quantified those wake-up signals to show that the performance of this technology is also dependent on the levels of interference in a noisy environment.

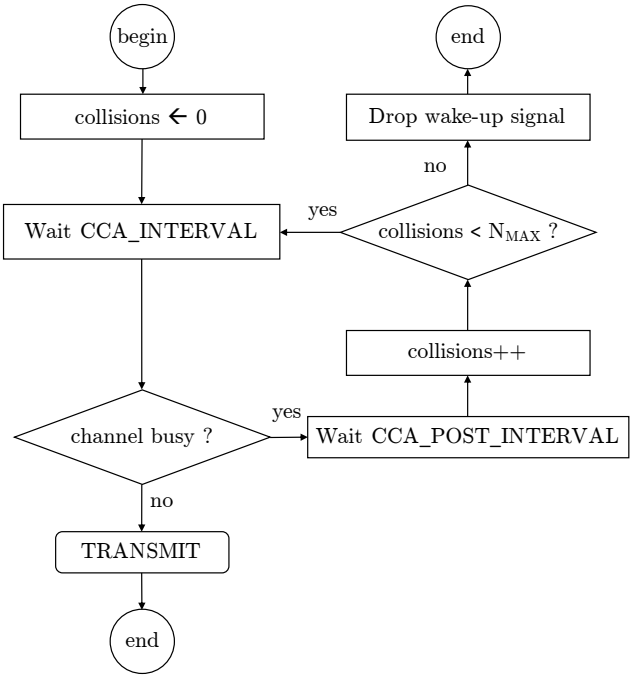


Figure 4.1: Transmission attempt flowchart with Clear Channel Assessment

The need for CCA capabilities has been already stated in previous works [29], [41]. However, no implementation has been done so far to take advantage of the already developed hardware and provide a CCA module in any WuR prototype. To best of our knowledge, the work presented in this chapter is the first analysis of interference patterns of a real WuR prototype with CCA capabilities in a noisy environment.

4.2 Clear channel assessment (CCA)

The CCA function takes advantage of the preamble detector module in the WuR circuit to sense activity in the channel. If a signal strong enough is received, this module generates an interrupt and turns on a *busy-channel* flag. The algorithm to transmit a wake-up signal is illustrated in Fig. 4.1. Whenever the transmitter wants to send a wake-up signal, it first waits for *CCA_INTERVAL* (set to 1 ms) and checks during that period if this flag turns on. In the negative case, it just sends the wake-up signal. Otherwise the channel is considered as busy, so it waits for a fixed amount of time called *CCA_POST_INTERVAL*, turns off the flag and restarts the procedure.

If the channel is busy due to another node transmitting a wake-up signal on the WuR, *CCA_POST_INTERVAL* should be set to the time required to transmit a wake-up signal. However, if the channel is busy due to noise or external interference, then this duration should be shorter to reduce the transmission delay. As the source of a busy channel can not be known in advance, we set the *CCA_POST_INTERVAL* to the time required to transmit half the length of a wake-up signal as a tradeoff between delay and the number of performed CCA (nu-

merous CCA attempts result in dropping wake-up signals). Notice that we can not reduce *CCA_POST_INTERVAL* to zero because the channel might be considered as free while a node is transmitting a sequence of zeros, due to the OOK modulation.

In the rest of this chapter, we are going to use the words *noise*, *interference*, and *collisions* as the same concept. Since the noise comes from a real environment, we do not know exactly the source of it. It can be electrical noise, thermal noise, collisions with other wake-up signal, interference with other RF signals, and so on. Special instruments, like a spectrum analyzer, may be used to reveal the source of the noise. In this work, we remained agnostic of the medium and did not use any of those instruments.

4.3 Lifetime model

The main purpose of WuR is to reduce the energy consumption of radio communication in wireless sensor applications. We propose here a set of equations that are necessary to estimate the battery lifetime of a generic node that transmits λ_{tx} and receives λ_{pkts} wake-up signals per second in mean values on the WuR channel. For simplicity, we do not consider transmission over the main radio. Moreover, we do not consider any specific communication protocol, apart from the use of the CCA algorithm.

The lifetime of an end-device depends on the effective battery capacity $C_{battery}$ and the average current consumption $\bar{I}_{total}$ drained by the application, and can be estimated with Eq. 4.1 [42].

$$Lifetime \approx \frac{C_{battery}}{\bar{I}_{total}} \tag{4.1}$$

The overall current consumption of the WuR module is composed of the part spent in each state of the application, which can be reduced to transmission attempt, reception, and idle.

$$\bar{I}_{total} = \bar{I}_{st\_rx} + \bar{I}_{st\_tx} + \bar{I}_{st\_idle} \tag{4.2}$$

In particular, the energy spent in the reception state can be written as

$$\bar{I}_{st\_rx} = I_{rx}\alpha_{rx} + I_{cpu}\alpha_{cpu} \tag{4.3}$$

where I_z is the instantaneous current consumption when the device is in power mode z, and α_z is the fraction of a second spent in power mode z (it is non-dimensional). However, in the WuR circuit the variation in the current consumption when it is receiving an active signal is insignificant, so I_{rx} and I_{cpu} are approximately the same, $I_{rx} \approx I_{cpu}$.

$$\bar{I}_{st\_rx} = I_{rx}(\alpha_{rx} + \alpha_{cpu}) \tag{4.4}$$

Consequently, the sum $\alpha_{rx} + \alpha_{cpu}$ is essentially the fraction of a second spent on the receiving state α_{st_rx}.

$$\bar{I}_{st\_rx} = I_{rx}\alpha_{st\_rx} \tag{4.5}$$

which can be estimated by Equation 4.6:

$$\alpha_{st\_rx} = \Delta t_{read}\lambda_{pkts} \; + \Delta t_{invalid}\lambda_{invalids} \tag{4.6}$$

where Δt_{read} is the time spent reading a wake-up signal, λ_{pkts} is the mean rate of received wake-up signals per second, $\Delta t_{invalid}$ is the time spent processing a signal with an invalid preamble and finally $\lambda_{invalids}$ is the mean rate of received signals with an invalid preamble per second. More specifically, the time spent on reading a wake-up signal is the sum of the length of the preamble, the length of the wake-up signal and the additional time the MCU requires to process the data.

$$\Delta t_{read} = \Delta t_{preamble} + \Delta t_{pkt} \ + \Delta t_{cpu\_rx} \tag{4.7}$$

The WuR receives and reads both *positive* wake-up signals (the ones that were transmitted by another node of the network) and *negative* wake-up signals (those that were produced from noise, interference or another network)

$$\lambda_{pkts} = \lambda_{positives} + \lambda_{negatives} \tag{4.8}$$

And in particular, for the positives, it only receives a fraction of the ones transmitted by the sender, depending on the reception success ratio (PDR) ρ:

$$\lambda_{positives} = \lambda_{tx}\rho \tag{4.9}$$

On the other hand, the average current consumption in the transmission state can be calculated with

$$\bar{I}_{st\_tx} = I_{tx}\alpha_{tx} + I_{cpu}\alpha_{cpu\_tx} \tag{4.10}$$

$$\alpha_{tx} = (\Delta t_{preamble} + \Delta t_{pkt})\lambda_{tx} \tag{4.11}$$

$$\alpha_{cpu\_tx} = (\bar{N}_{cca}\Delta t_{cca} + \Delta t_{cpu\_tx})\lambda_{tx} \tag{4.12}$$

where $\bar{N}_{cca}$ is the mean number of transmission attempts due to the CCA algorithm detecting a busy channel, and Δt_{cca} is the time spent in each CCA cycle, which corresponds to $\Delta t_{cca} = CCA\_INTERVAL + CCA\_POST\_INTERVAL$.

Finally, we assume that the device sleeps in low power mode during the idle state, so the average current consumption spent in that case is the sleep mode current scaled by the fraction of a second spent on this state:

$$\bar{I}_{st\_idle} = I_{sleep}(1 - \alpha_{st\_rx} - \alpha_{tx} - \alpha_{cpu\_tx}) \tag{4.13}$$

In order to analyze the contributions of the useful radio communication and the noise, we can decompose the current consumption of the reception state into the contribution of the actual protocol and the one of the noise, based on equations 4.5, 4.6 and 4.8.

$$\bar{I}_{st\_rx} = \bar{I}_{rx\_protocol} + \bar{I}_{noise} \tag{4.14}$$

$$\bar{I}_{rx\_protocol} = I_{rx}\Delta t_{read}\lambda_{positives} \tag{4.15}$$

$$\bar{I}_{noise} = I_{rx}(\Delta t_{invalid}\lambda_{invalids} \ + \Delta t_{read}\lambda_{negatives}) \tag{4.16}$$

Then, the total current consumption can be re-written as:

$$\bar{I}_{total} = \bar{I}_{rx\_protocol} + \bar{I}_{st\_tx} + \bar{I}_{noise} + \bar{I}_{st\_idle} \tag{4.17}$$

and finally, aggregating the components related to the communication protocol:

$$\bar{I}_{protocol} = \bar{I}_{rx\_protocol} + \bar{I}_{st\_tx} \tag{4.18}$$

$$\bar{I}_{total} = \bar{I}_{protocol} + \bar{I}_{noise} + \bar{I}_{st\_idle} \tag{4.19}$$

4.4 Experimental platform

The network is composed of three nodes: one sink and two transmitters referred to as transmitter 1 and transmitter 2. The sink is only receiving and logging every received signal to a computer. A 2-bytes length signal is considered as wake-up signal when the preamble is correctly received. The preamble is composed of 3 bits, and its value is "110". Signals with invalid preamble are categorized as *Invalids* and are discarded by the sink. Transmitters are sending a 2-bytes length wake-up signal every second at 1 kbps. The content of each wake-up signal is respectively set to 0xAA1A and 0xA1AA for transmitter 1 and transmitter 2. Wake-up signals whose content matches the expected content (0xAA1A or 0xA1AA) are referred to as *Positives*. On the contrary, *Negatives* are wake-up signals whose content differs from the expected content. In addition, transmitter 2 uses our CCA module while transmitter 1 does not.

All nodes are using a 3.6 dBi gain antenna (ANT-868-CW-RCS [43]). For availability purposes, the transmitters are fed with a 3 V supply power (2 AAA batteries), while the sink is fed with 3.3 V, powered by an Arduino UNO that acts as a bridge between the WuR prototype [44] (through I2C communication) and the computer (through UART communication). Our preliminary experimentation campaign, performed in a controlled environment with no interference, showed that we can achieve a 100% packet delivery ratio with no *Negatives* (being *True Negatives* or *False Negatives*) or *False Positives* when the transmitter and the receiver are separated by a distance below 1 m. For the present campaign, we placed the transmitters and the sink at a distance of 70 cm to compare the results with that ideal case. Such a short distance reflects scenarios involving a very dense network and enables transmitters to experience the same interference. The experiment was performed indoors, next to a window in a regular office at the ICube Laboratory of the University of Strasbourg as depicted in the Fig. 4.2.

To measure the instantaneous current consumption of the device in each mode (I_{tx}, I_{rx}, I_{sleep} and I_{cpu}) we load specific pieces of firmware to keep the device under test (DUT) in a single mode continuously. The current measurements were taken with the multimeter FLUKE 177 True RMS in DC mode. For the sleep mode, the multimeter was used in the DC voltage mode to measure the voltage drop in a 1 kΩ resistor added in series with the power supply. In all cases, the measured value has an error between 1% and 2%, given the specified precision of reading and number of least significant digits of the multimeter, as well as the error propagation for the sleep current case. Notice that in the TX mode, the DUT was transmitting

Figure 4.2: Experimental platform

continuously a sequence of bits set to '1' at +10 dBm and 868 MHz, so it is the worst case of maximum current consumption in that mode. In the RX mode, the DUT is continuously waiting for a reception interrupt and a transmitter (transmitting continuously a sequence of bits set to '1') was placed close to it. The CPU mode used an infinite loop incrementing a dummy variable with all the required peripherals turned on (mainly the timers and the I2C module) and no compiler optimizations. Finally, the sleep mode simply sets the DUT in low power mode. In each case the current consumption was measured for the whole board which includes the WuR receiver circuit as well as the ultra-low-power MCU circuit.

Finally, we measured the real value of all Δt variables defined in Section 4.3. Measurements were taken signaling a pin on the MCU and measuring timing differences with the cursors of a digital oscilloscope Tektronix TBS1104.

4.5 Results

We performed numerous experimentations and all collected results were very similar. For clarity reason, results presented in this section are extracted from one experimentation, corresponding to a duration of 24 hours, which is why standard deviation or confidence intervals can not be calculated.

4.5.1 Packet delivery ratio

Fig. 4.3 shows the PDR achieved by each transmitter together with the number of *Negatives* and *Invalids* received per second. *Negatives* are wake-up signals that were received but their contents differ from the original content (i.e. differ from 0xAA1A and 0xA1AA) while *Invalids* are signals with invalid preambles. We can see that there is a high correlation between the dynamic of the noise and the behavior of the PDR for both transmitters. During quiet moments, i.e. when the noise is low, we see that both transmitters achieve their maximum PDR (e.g. 95% for transmitter 1 and 97.5% for transmitter 2 around 7h00). Transmitter 2, with CCA, has better performance, though. This can be explained if we notice that both transmitters synchronize their transmission phase every approximately 15 mins because of the clock drift, as illustrated in Fig. 4.4. During each synchronization period, the PDR of transmitter 1 experiences severe drops reaching down to 25% while transmitter 2 maintains a PDR higher than 85% thanks to our CCA algorithm.

Moreover, transmitter 1 is more sensitive to noise peaks. When the noise level is higher, for example around 18h00, we see that the PDR of transmitter 1 decreases to 75%, while transmitter 2 maintains a PDR higher than 82%. This means an

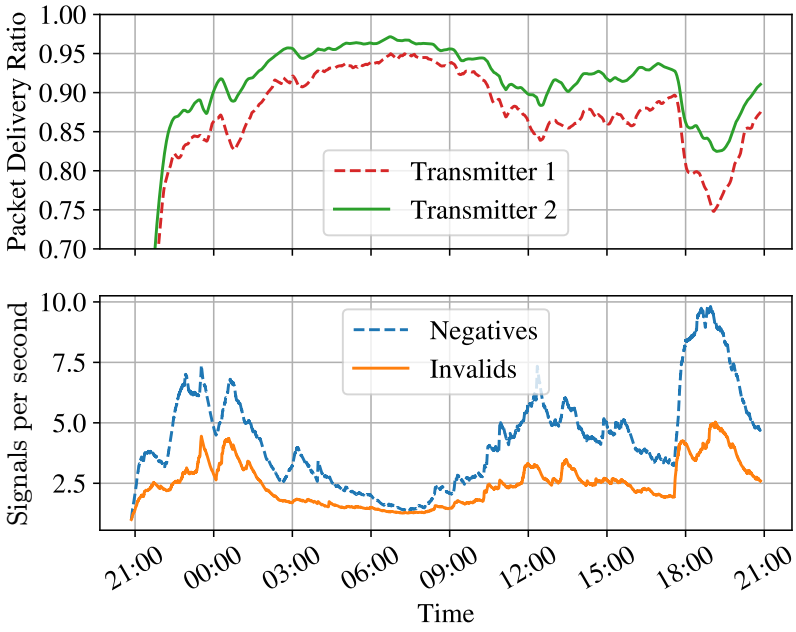


Figure 4.3: Packet delivery ratio and interference

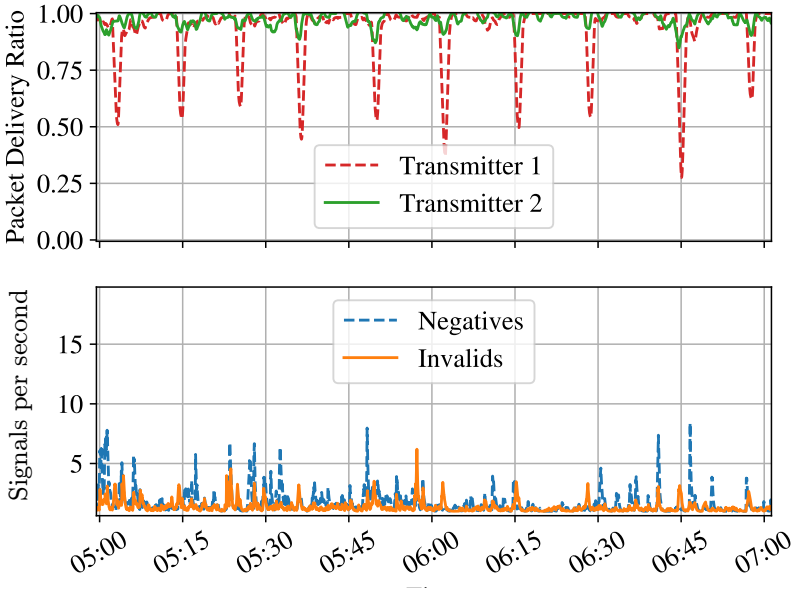


Figure 4.4: Transmission phase synchronization because of the clock drift

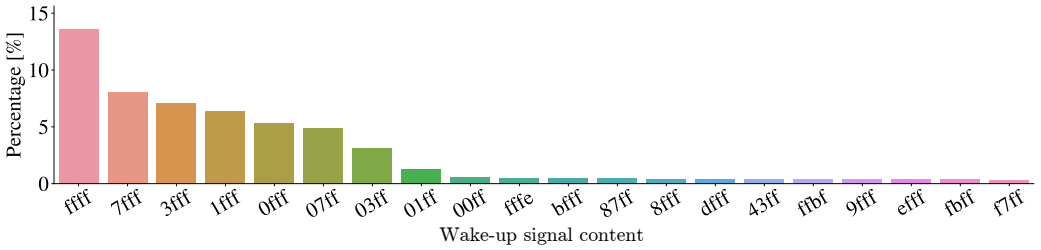


Figure 4.5: Negative wake-up signals histogram

improvement of 10% when using CCA in an environment of $\lambda_{negatives} = 10$ and $\lambda_{invalids} = 5$.

4.5.2 Wake-up signal contents

Fig. 4.5 shows the count of the 20 most significant *Negatives* received by the sink, which represent 55% of the total number of *Negatives*. We can see that most of *Negatives* are long sequences of bits set to '1', namely 0xffff, 0x7fff, 0x3fff, and so on. This can be explained considering the modulation used in other wireless communication technologies where the carrier frequency is always-on during the frame transmission. For example, in frequency shift keying or phase shift keying, the signal is always-on during the frame transmission and the variations in frequency or phase respectively represent the different symbols. Since the WuR circuit is an envelope detector for OOK it can not distinguish those variations. Instead, it simply translates an on signal to a bit set to '1' and an off signal to a bit set to '0'. We are therefore convinced that the main source of *Negatives* are external wireless communication technologies.

Comparatively, if we look at the 20 closest *Negatives* to the original content (0xAA1A and 0xA1AA), they respectively represent 0.26% and 0.1% of the total number of *Negatives* for transmitter 1 and transmitter 2. Those wake-up signals were mostly generated by collisions, inverting some bits or just shifting the whole content. Thanks to CCA, we can see that transmitter 2 is less prone to collisions.

4.5.3 Current consumption

The instantaneous currents measured with the multimeter for each mode are presented in Table 4.1. On the other hand, the timing measurements done with the oscilloscope are shown in Table 4.2. Finally, the equations of the current consumption model presented in Section 4.3 were computed with the values of Table 4.3.

Fig. 4.6 shows the current consumption contribution of each term present in Equation 4.19 over a range of wake-up signal generation period (T). For low traffic scenarios ($T > 45$ s), we can see that the idle state contributes more to the current consumption than the effective radio communication ($\bar{I}_{protocol}$). However, when the environnement is noisy, the receiver experiences many false wake ups and $\bar{I}_{noise}$ becomes the main source of current consumption. This means that, for very low traffic applications, improving further the communication protocol stack can only marginally increase the lifetime of end-devices. The T threshold where $\bar{I}_{protocol}$

Table 4.1: Current consumption measurements for V = 3 V

Mode	Current [mA]
I_{tx}	14.3
I_{rx}	0.40
I_{sleep}	0.00761
I_{cpu}	0.40

Table 4.2: Timing measurements

Variable	Value [ms]
$\Delta t_{preamble}$	3
Δt_{pkt}	16
Δt_{cpu_rx}	3
Δt_{cpu_tx}	4.4
Δt_{cca}	10
$\Delta t_{invalid}$	4

Table 4.3: Parameters for modeling the current consumption

Variable	Value
$\lambda_{invalids}$	0-2 signals per second
$\lambda_{negatives}$	0-2 wake-up signals per second
λ_{tx}	1/100 - 1/10 wake-up signals per second
ρ	0.9
$\bar{N}_{cca}$	10

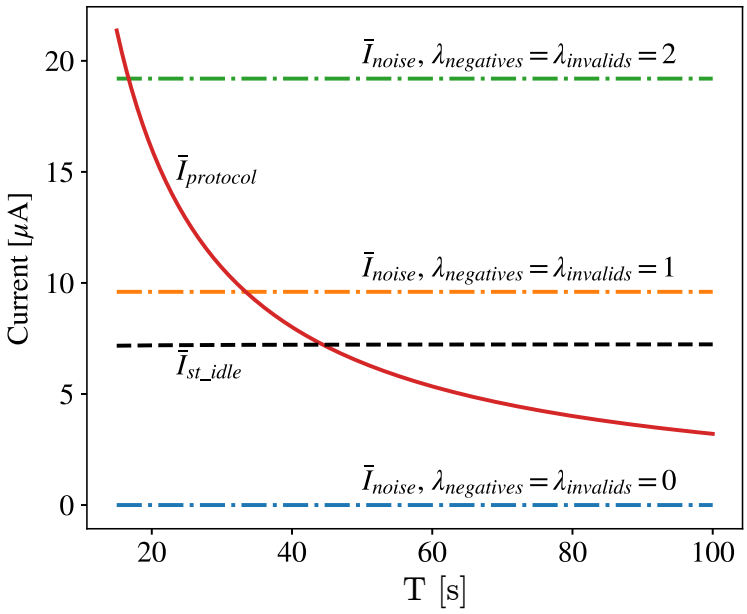


Figure 4.6: Current consumption contributions

remains the main contributor to the current consumption depends on the amount of noise ($\lambda_{negatives}$ and $\lambda_{invalids}$). As the level of noise goes up, the threshold goes down. Said differently, $\bar{I}_{protocol}$ can only be greater than $\bar{I}_{noise}$ when $\lambda_{positives} >> \lambda_{negatives}$.

4.6 Conclusions

In this chapter, we presented a performance study of WuR in a real-world environment. An experimental platform for a proof of concept was deployed to log data from a WuR-based receiver into a computer for further analysis and post-processing. We implemented a software module to perform CCA without further modifications of the hardware. In Section 4.5 we showed that this module improves the performance of the communication in noisy environments and reduces the number of collisions, contributing to increase the overall packet delivery ratio. In particular, it can improve the PDR from 25% up to 85% for internal interference.

We also analyzed the behavior of external interference and how they affect the performance of the WuR technology. We categorized errors as *Negatives* - packets received with a wrong content and *Invalids* - signals with an invalid preamble. The results showed that external interference mainly generate *Negatives* composed of a sequence of bits set to '1'. Using line coding scheme like 4B5B would force transitions in legacy signals and therefore would help to discard such erroneous wake-up signals.

Finally, a current consumption model was presented and evaluated based on the measurements we performed on a real prototype and the outcomes of the interference experiments. Our analysis throws new insights on the WuR behavior: in low traffic scenarios, optimizing the communication protocol stack will only marginally increase the lifetime of the end-devices. The energy spent on false wake ups due to ambient

noise or the minimum energy spent on the idle state can be more significant than the average energy consumed by the communication protocol. However, the number of false wake ups can be reduced by using early sleeping techniques [45].

As a future improvement, the experiment could be carried out through longer periods (weeks instead of days) and at a larger scale (in terms of distance and number of end-devices). Moreover, adding a proper electromagnetic shield would make the prototypes less prone to external interference. Also, a more complex test could be carried out by connecting the transmitters to the computer to log their data. This way, we would have more flexibility to control the data transmitted and differentiate exactly *Negatives* due to noise and the ones due to collisions. A random packet generation process could also be used to reduce the synchronization effect because of the clock drift between the transmitters. Finally, with a longer preamble, the firmware should be ready to communicate at 5 kbps (instead of 1 kbps), and with some code optimizations it might scale up to 10 kbps. Such throughput could improve the overall WuR performance by reducing the transmission delay and the contention in the medium.

Summary of the first contribution

The analysis of the MAC layer of WuR networks concludes in this chapter. One of the main ideas to take away from this part is that WuR-based solutions are very efficient compared to duty-cycled MAC protocols in terms of latency and power consumption if we limit the number of nodes relaying the wake-up signal to 3 or up to 5. Regarding the sync delay problem, we found that for unreliable medium the best option is to use a fixed sync delay, instead of a dynamic one sized by the WuS-ACK mechanism.

We understand that the WuR packet retransmissions have a minor impact on the power consumption and that the PDR is heavily affected by the WuR medium reliability. Accordingly, we could just increment the maximum number of retransmissions for the WuR, while keeping the same parameters in the main radio. This way we would have more chances to deliver the WuR packets, countering the medium reliability with a minor impact on power consumption. For this to happen, we must add a new mechanism for each relay node to find out if the next relay in the path has received the packet. This can be easily implemented by overhearing the WuR medium. This will also bring a tradeoff between PDR and latency, as the increased number of retransmissions comes by the hand of higher latency, as we saw in the results with collisions of Section 3.4.

Furthermore, by inspecting the interferences in the WuR channel we found that it is possible to provide better noise immunity by using a coding scheme that forces transitions. And finally, we explained that in low traffic applications, the communication protocol does not impact strongly the battery lifetime of the device. Consequently, the WuR technology is especially interesting in low traffic scenarios.

Up to now, we have used a static route to relay wake-up signals to the final destination. However, in real applications, we need to compute that path dynamically. In addition, in traditional WSN networks, the next hop at the routing layer is the same node as that of the MAC layer. In the next part, we are going to explore the benefits of WuR at the routing layer. First, we propose a cross-layer solution using WuR and based on existing infrastructure, in order to opportunistically find the best hops in a routing path. Then, we address the challenge of designing a routing protocol considering the range mismatch.

Part II

Second contribution: A new routing layer solution for WuR

Chapter 5

LoBaPS: Load Balancing Parent Selection for RPL

Contents

5.1 Introduction

In our first steps in this part (chapters 5), we are going to limit the range of the main radio to match that of WuR and overcome the sensitivity mismatch described in Section 2.4. That way, we are going to be able to see the impact on an existing routing infrastructure when using WuR at the MAC layer. However, reducing the

main radio range might not be feasible in the real world because the transceivers are normally optimized to be used at the maximum output power. Furthermore, it increases the number of hops that are required to deliver a data packet, which decreases the performance in many ways. Then, in the following steps (chapter 6), we are going to remove this hypothesis and start designing a new routing infrastructure that addresses the range mismatch.

The Routing Protocol for Low Power and Lossy Networks (RPL [12]), is the Internet Engineering Task Force (IETF) standard for calculating routes in multi-hop WSN. In RPL, the network is a Destination Oriented Directed Acyclic Graph (DODAG), where the sink is the root. One of the key features of this protocol is the rank of each node. The rank is a level of how far away a node is from the sink. In order to establish it, the nodes exchange control messages (DIO) advertising its information. To calculate the rank, RPL uses a metric, for example, ETX or the minimum amount of hops (MinHop), and an Objective Function, which translates the metric into the rank value. A commonly used Objective Function is OF0 [46] because of its stability and simple implementation. OF0 selects as the preferred parent the one with the best metric and a backup feasible successor.

However, RPL still presents some open problems: inefficient parent selection, slow recovery time after a preferred parent dies and energy bottleneck (the preferred parent consumes way more energy than the rest of its siblings limiting the lifetime of the network). ETX metric is a well-known solution to select efficient parents for reliable routes, but it presents serious issues with stability because of the recurrent parent changes [47]. Also, it is difficult to compute and maintain this metric in RPL where the frequency of the generation of DIOs is reduced over time, while the data traffic may increase. In contrast, MinHop is very stable, but might use routes with bad quality links [47]. In addition, the underlying duty-cycle in the MAC layer increases the latency in an effort to reduce the energy consumption, thus limiting the performance for high traffic loads.

Consequently, this contribution investigates how WuR can address these issues at the MAC layer, relying on an existing routing infrastructure. The main idea is to allow the source node of a data packet to wake up multiple feasible forwarders thanks to the always-on feature of WuR (it is always listening to the WuR channel). Then, all those nodes compete to forward the data packet by using a backoff period before the retransmission.

Contribution

- We present LoBaPS, an approach to combine the best of both worlds: the power efficiency and always-on feature of WuR with the stability of OF0 and MinHop in RPL.
- Moreover, we put the focus on load balancing in order to extend the lifetime of the network.
- Then, we present Energy LoBaPS (eLoBaPS), an improvement over LoBaPS that takes a step forward into the ideal energy balancing in WSN. In this protocol, the backoff period is proportional to the battery consumed by each parent. In addition, the nodes that are consuming a lot stop competing for a while and let other feasible forwarders spend their batteries.

Publications:

- S. L. Sampayo, J. Montavont and T. Noel. LoBaPS: load balancing parent selection for RPL using wake-up radios, in the 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, July 2019, rank B.
- S. L. Sampayo, J. Montavont and T. Noel. Selecting Parents with Wake-Up Radios for Load Balancing in RPL, in the 4ème rencontres francophones sur la conception de protocoles, l'evaluation de performance et l'expérimentation des réseaux de communication (CoRes), Saint Laurent de la Cabrerisse, France, June 2019.
- S. L. Sampayo, J. Montavont and T. Noel. eLoBaPS: Towards Energy Load Balancing with Wake-Up Radios for IoT, in the 18th International Conference on Ad Hoc Networks and Wireless (AdHoc-Now), Luxembourg, Luxembourg, October 2019, rank B.

5.2 LoBaPS

The main contribution of this chapter is the Load Balancing Parent Selection (LoBaPS) protocol that takes advantage of the Wake-Up Radio (WuR) to select opportunistic parents in RPL. LoBaPS starts operating once RPL has converged and only supports convergecast data traffic. In this work, we remind that we are assuming that the WuR range is the same as that of the main radio range.

5.2.1 Concept

The source of the application packet initiates the communication by transmitting a packet over the WuR channel, called Wake-Up Request (WREQ), which contains the

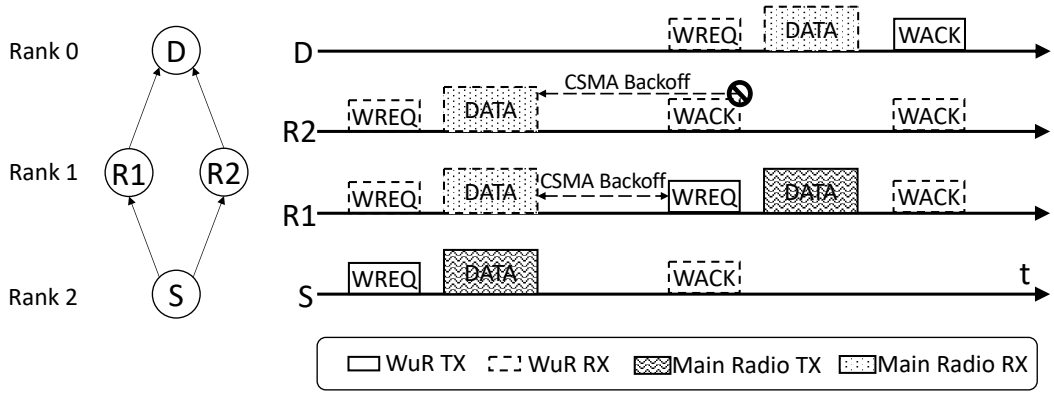


Figure 5.1: Example of the algorithm in a timeline

node's own rank together with a unique application ID, as depicted in Fig. 5.1. All nodes in the vicinity of the sender will receive this WREQ as they are continuously listening to the WuR channel. Whenever a node receives a WREQ, it compares the received rank with its own rank, and only wakes up its main radio if the former is higher than the latter. This way, only nodes with lower rank (which are closer to the sink) can forward the packet, avoiding routing loops.

A short time after transmitting the initial WREQ, the source sends the data packet over the main channel, turns off its main radio, and starts a timer to wait for the acknowledgment over the WuR channel. When the sink (which is the final destination of all data packets) wakes up its main radio and receives a data packet, it sends back an acknowledgment via the WuR channel. In the case of an intermediate node, it tries to forward it upwards by transmitting a new WREQ with its own rank, such as node R1 in Fig. 5.1. The purpose of this WREQ is threefold: to wake up next hops toward the sink and to acknowledge data reception for the sender (the third purpose is detailed in Section 5.2.2). As a result, an acknowledgment (WACK) only differs from the WREQ that triggered its transmission by the fact that the advertised rank is lower than the one included in the WREQ.

5.2.2 WREQ collisions

Due to the nature of the wireless medium, a single data packet may be received by more than one parent (cf. R1 and R2 in Fig. 5.1). To limit collision, the CSMA layer of each forwarding node calculates a random backoff period before the transmission of the new WREQ. The node for which the backoff expires first will send a WREQ, cancelling the ongoing backoff of the other forwarders. This random backoff ensures that the feasible successors do not try to retransmit the packet at the same time generating collisions.

Collision on initial WREQ can also occur, especially when the WuR works at low data rates, because the time over the air is significant and can be longer than that of the main data. In consequence, the channel is extremely sensitive to collisions because the transmission opportunities are very limited, as it was shown in Chapter 3. Thus, a Clear Channel Assessment (CCA) function is mandatory to improve significantly the reliability in real environments, as concluded in Chapter 4.

Although CCA is very common in traditional radio transceivers, we are part of the only few proposals investigating its usage in WuR [30]. As a consequence, a CCA module is implemented in the WuR driver and is used every time a message is transmitted over the WuR channel. When the WuR channel is sensed as busy, a collision error is passed to the CSMA layer. We are convinced that such a feature is required to increase the overall network performance as supported by the results presented in Section 5.6.

5.2.3 Cross-talk

A problem that might arise using LoBaPS is the cross-talk, that is, when a WREQ is misread as a WACK. This can happen when multiple nodes initiate a communication at the same time, or during an ongoing communication. In order to overcome this problem, we use a unique application ID in every WREQ. This ID is set by the original source and kept unmodified by intermediate nodes until reaching the sink. Furthermore, whenever a node initiates a new communication, it uses a different application ID than the previous one. Therefore, a relay can distinguish between a new WREQ or the WACK of one of its previous transmission. For our implementation, we assign a batch of unique application IDs that can be used for each node to start a communication.

5.2.4 Retransmissions avoidance

Reducing duplicated packets as much as possible is necessary to achieve efficient load balancing. If a source would miss an acknowledgment, then the CSMA layer would try to transmit the packet again waking up all its feasible successors one more time even if one of them have already forwarded it. In order to reduce this problem, each node keeps a list of recently viewed application IDs and a list of forwarded application IDs. The first list is intended to avoid waking up the main radio and waste power listening to a packet that has already been handled by another node. On the other hand, the second list keeps track of the application IDs that have actually been transmitted by this node. We remind that the WuR module is always on listening to the WuR channel thanks to its ultra-low power consumption. Whenever a node receives a WREQ with an application ID that is currently in this list, it will immediately transmit a WACK with its own rank to let the source know that this packet has already been handled and that it must refrain from retransmitting it. We call this specific WACK a WuR Duplicated ACK (WDA). Both lists are cleaned periodically to avoid outdated values.

In addition, when the winner of the forwarding competition (the node with the shortest backoff period) transmits the WREQ, the rest of the competitors overhear this signal and refrain from retransmitting, thus avoiding duplicated packets.

5.3 eLoBaPS: Improved load balancing

The main problems with LoBaPS are the way in which the load is balanced and the energy wastage in listening mode. The load is balanced randomly, so it is not the optimal solution towards energy efficiency. On top of that, there is a significant

amount of energy wasted in listening mode when all the feasible successors wake up their main radio, limiting the network lifetime. In this section, we present eLoBaPS, an improvement of LoBaPS that increases the network lifetime and provides better energy balancing. In this extension, the general behavior is also described by Fig. 5.1. In this case, the backoff period is proportional to the energy consumed by the node, so that nodes with more remaining battery have more chances to win the competition.

The first approach in order to calculate a backoff period aware of the energy consumption is to make it directly proportional to the amount of consumed battery percentage:

$$B_j(t) = Ke_j(t) + C \tag{5.1}$$

where $e_j(t)$ is the energy percentage consumed by the node j at time t, and K is a constant parameter to adjust the units to milliseconds. A small random contention window C uniformly distributed with range $[0, T_c]$ is added to mitigate the case where more than one node has the same amount of energy. However, the battery discharges as the time goes by, so $e_j(t)$ is proportional to t. In consequence, the backoff period will increase over time as the battery discharges. Notice that the backoff period adds latency to the protocol, so it would increase the end-to-end delay of the application. In order to keep it stable as the battery discharges, we came up with a slight modification of Eq. 5.1:

$$B_j(t) = K[e_j(t) - e_{dj}(t)] + C \tag{5.2}$$

where $e_{dj}(t)$ is the desired energy consumed at time t. This variable is such that if all the relays consume this energy, the load is balanced and the network lifetime is maximized. In order to estimate it, the nodes include their energy consumption in all the packets sent on the WuR channel (WREQs and WACKs) so that they overhear the current energy of all their neighbors. Then the value is estimated for node j with a metric r as:

$$e_{dj}(t) = \min_{i \in R}\{e_i(t)\} \tag{5.3}$$

where R is the set of all nodes i with a metric of r.

Although this may balance the energy by controlling the transmissions, it is not reducing the energy consumed by the listening mode every time a node requests several relays to wake up and listen to the packet (see node R2 in Fig. 5.1). With this in mind, another feature is included in the algorithm to reduce the listening mode energy: if the current energy consumed by the node is above a certain threshold on top of the desired energy consumption, then the node does not wake up and listen to the main radio channel whenever it receives a WREQ. However, this feature can create problems when a node is the only parent possible for some nodes for example. In such a situation, the packet will be delayed until the parent saves enough energy to keep up with $e_d(t)$. For these reasons, this threshold should be chosen carefully to not degrade the performance. In this work, it has been set to the energy consumed by the node that has consumed the most among all its neighbors in R, that is:

$$e_{threshold j}(t) = \max_{i \in R}\{e_i(t)\} \tag{5.4}$$

Fig. 5.2, Fig. 5.3, Fig. 5.4, and Fig. 5.5 may help to understand the overall behavior of the protocol. Fig. 5.2 shows the overall finite state machine of a generic

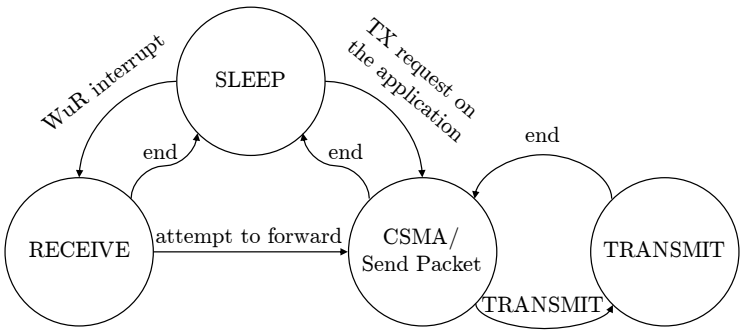


Figure 5.2: Finite state machine of the entire system

node running the eLoBaPS protocol. Besides, Fig. 5.3, Fig. 5.4, and Fig. 5.5 describe the internals of each mode. When the application layer of the node wants to send a message, it issues a transmission request to the lower layers. The CSMA layer receives this request and performs the algorithm described in Fig. 5.3, calling the routine of the TRANSMIT mode, detailed in Fig. 5.4. Notice that the COLLISION output of the TRANSMIT routine is a flag to prevent from a collision. In such a case, the TRANSMIT routine does not get to transmit the DATA on the main radio. This is the classic behavior of the CSMA protocol. Its implementation in ContikiOS was not significantly modified. The values of the constant parameters used in our implementation can be found in Table **??**. On the other hand, when a node receives a WREQ, an interrupt is triggered and the RECEIVE mode is activated, following the steps in Fig. 5.5.

5.4 Optimized W-MAC

To evaluate the performance of our protocols, we implemented W-MAC, a reference protocol that uses WuR, which was introduced in Section 2.3, and supports RPL in its traditional way.

This protocol is based on W-MAC [17], a straight forward utilization of the WuR, already described in Section 2.3. In this work, the parameters of this protocol have been optimized (sync delay, reception window timeout, etc.) and a CCA capability has been introduced to avoid collisions. In addition, the length of the Wake-Up signal has been fixed to 2 bytes, which complies with the short address of 16 bits in IEEE 802.15.4. We use the default values in that standard for the CSMA algorithm on top of this layer.

This protocol uses RPL with Objective Function 0 and MinHop metric. In RPL OF0 RFC [46] there is no clear explanation on how to detect that the preferred parent of a node is no longer available. In the general implementations, when a preferred parent dies, the node does not realize it, and there is no mechanism to trigger a parent change. However, the RPL OF0 RFC provides a *backup feasible successor* that can be used whenever this happens. For this reason, we implement the parent change trigger when a fixed number (MAX_T) of communication attempts fail consecutively. Notice that a communication attempt includes all the retries at the CSMA layer. This means that if the maximum amount of retries at the

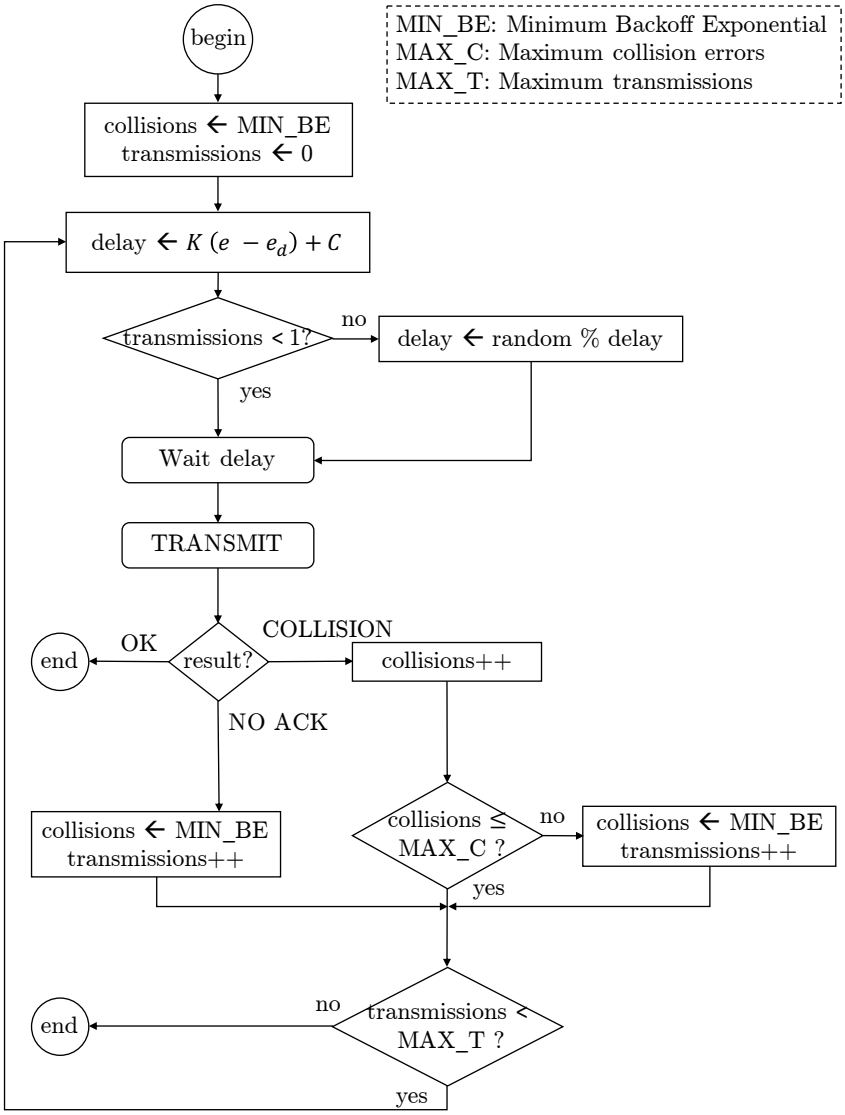


Figure 5.3: CSMA mode flowchart

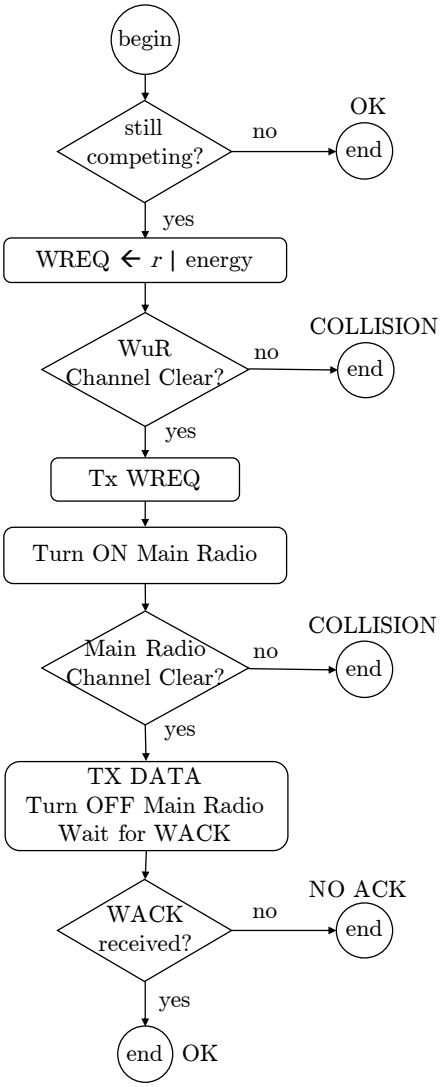


Figure 5.4: TRANSMIT mode flowchart

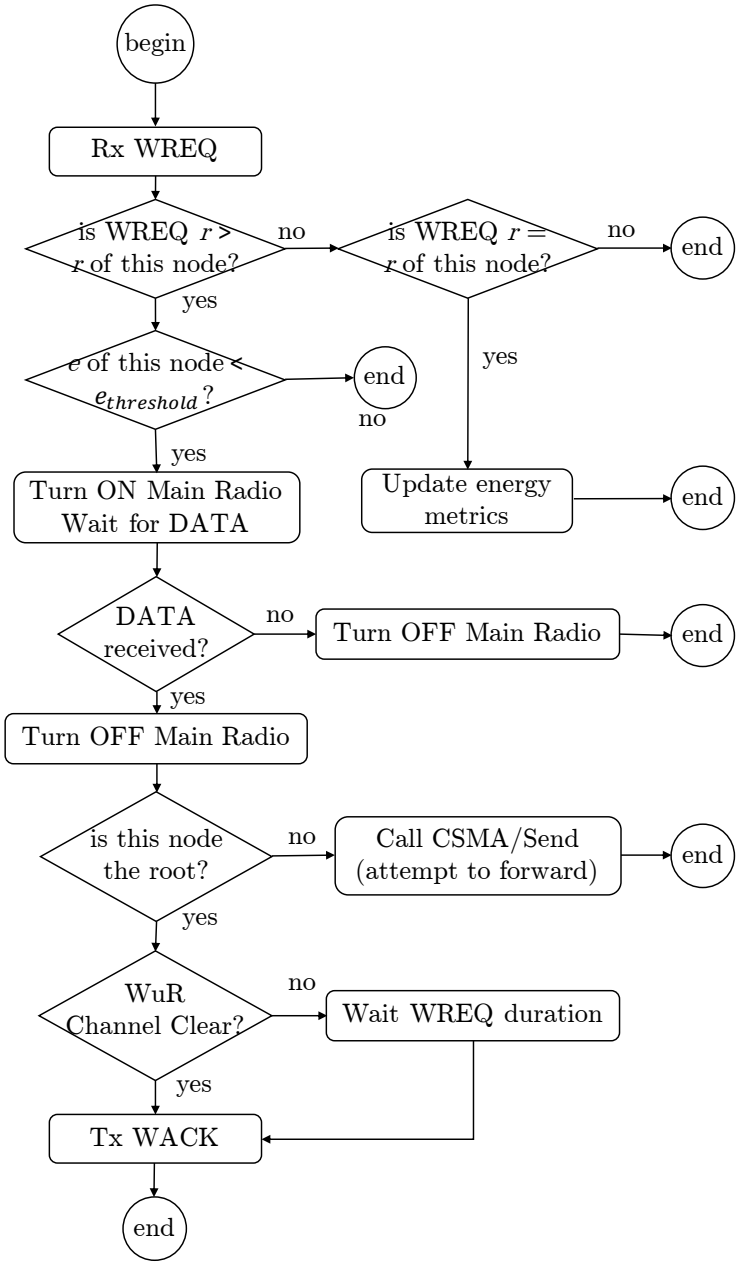


Figure 5.5: RECEIVE mode flowchart

CSMA layer is 3 for example, and the MAX_T is 4, then the parent change will be triggered after $3 \cdot 4 = 12$ acknowledgments not received consecutively. As soon as this happens the preferred parent is removed from the parent set and the backup feasible successor is used instead. Then, if at some point the backup parent also fails MAX_T times, the parent set is cleaned and a RPL local repair is issued.

5.5 Simulation framework

In this chapter, we use the same simulation framework that we used in the previous part of the thesis, based on WaCo [17], a COOJA extension. This system reproduces the actual firmware that runs on real devices. On top of that framework, we added some new features for this work.

5.5.1 Simulation setup

A summary of the simulation parameters is given in Table 5.1. We use the default values in ContikiOS for other parameters.

The energy consumption is calculated with the help of Powertrace. The electric values required are taken from the Sky Mote datasheet [36] and WaCo [17].

Additionally, the simulations are performed in a triangular grid topology as depicted in Fig. 5.6. Here, the nodes are at a maximum of 2 hops away from the sink and with a node density (i.e. number of nodes per unit area) such that each leaf can have between 2 and 7 feasible parents. In this figure, we can see the links, represented by arrows, of each node to all its feasible next hops. In order to simulate its battery lifetime, each node keeps track of the energy consumed and when it reaches some maximum level (defined as a parameter) the node is shut down. This value is defined so that the sink receives around 1000 packets when the first node dies. When this happens there are two different scenarios that follow. First, some children might be temporarily unreachable (in the case of W-MAC with a routing protocol that is based on a preferred parent), but after a repair mechanism is triggered , a new parent can be found and the network graph continues to be connected. Second, some nodes might be left far away from any other one, becoming absolutely unreachable, and no mechanism can get the network graph connected again. The simulation finishes immediately and exclusively when the second scenario is found. This way, it is possible to analyze the behavior of the network after the first node dies and throughout the process of parent changes.

We implemented LoBaPS, the extension eLoBaPS and W-MAC with RPL and discuss the results in the following section.

5.6 Results

5.6.1 Packet delivery ratio

The reliability is studied in Fig. 5.7 by analyzing the evolution of the Packet Delivery Ratio (PDR) over time for a 10 s IPI in mean values and with the 95% confidence interval. In addition, it is possible to see the decline of the PDR after the first node dies, which happens between 750 s and 1200 s for W-MAC, and around 1200

Table 5.1: Simulation parameters

Parameter	**Value**
Number of nodes	15
Repetitions of each simulation	100
MAC layer	CSMA (Contiki version)
CSMA minBE	3
CSMA maxBE	5
CSMA maxBackoff	4
CSMA maxRetries	3
Network layer	uIPv6
RPL Mode	No downward routes (MOP 0)
RPL Objective Function	OF0 [46]
RPL Metric	Minhop
eLoBaPS T_c	30 ms
eLoBaPS K	11.6 ms
MAX_C	7
MAX_T	4
Inter Packet Interval (IPI)	1, 5, 10, 60 s
WuR packet length	16 bits
WuR data rate	10 kbps
Main data packet length	80 bytes
Main data ACK packet length	5 bytes
Main radio data rate	250 kbps
Main Node	Sky mote [36]
WuR HW prototype	[16]
WuR Supply Voltage	1.8 V
WuR TX current	16 mA
WuR RX current	80 uA
WuR idle listening power consumption	1.944 μW
Main Radio medium model	UDGM
WuR Radio medium model	UDGMConstantLoss
Main radio RX success ratio	80%
WuR RX success ratio	80%

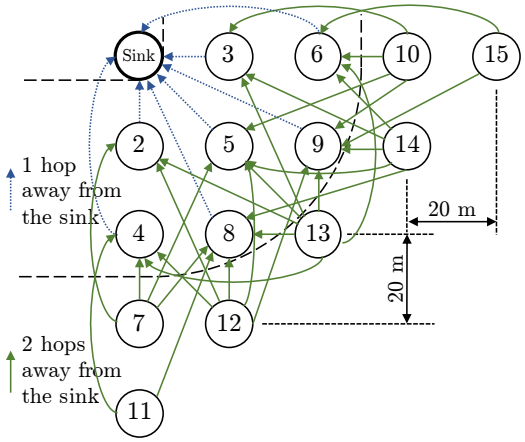


Figure 5.6: Test topology

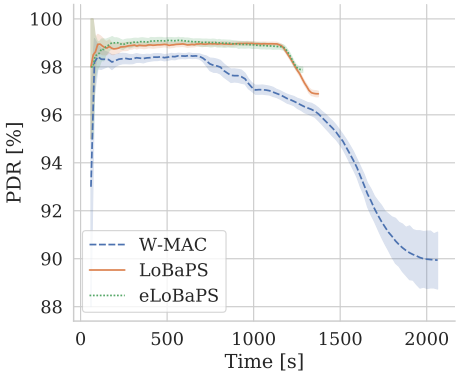


Figure 5.7: Packet Delivery Ratio for IPI = 10s

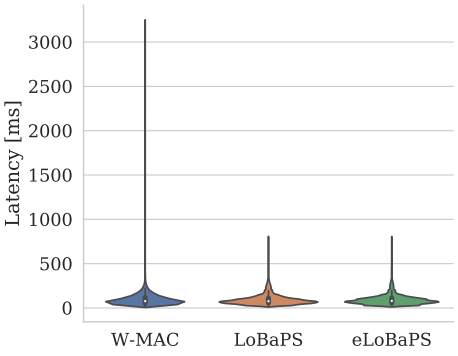


Figure 5.8: Latency for IPI = 10s

s in LoBaPS and eLoBaPS. In W-MAC protocol, the PDR decreases fast and with high variability after this point, while in LoBaPS there is good stability during the network lifetime and a precise and controlled decline slope. eLoBaPS improves this point by reducing the length of the decline, so there is a better ratio of stable time over decline time.

5.6.2 End-to-end latency

Fig. 5.8 shows the end-to-end latency as a violin plot that takes into account every successful packet transmission in all repetitions for an IPI of 10 s. In eLoBaPS, the latency is similar to the one achieved by LoBaPS, but in this case, it only depends on the number of retransmissions because there is no backoff exponential. The actual amount of time of the backoff period in eLoBaPS is adjusted so that it is similar to that of the average CSMA backoff period in LoBaPS. In this plot, we can also see that the lack of precision and the inefficient parent selection after a parent dies in W-MAC can lead to extremely high maximum values of latency, while in LoBaPS and eLoBaPS, the maximum is three times smaller.

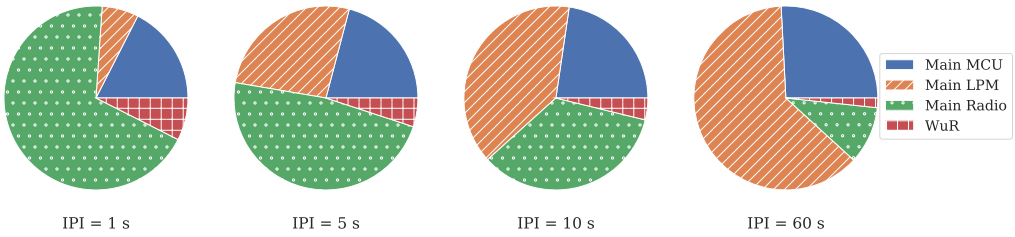


Figure 5.9: Energy profile for different IPIs for the W-MAC protocol

5.6.3 Lifetime

Fig. 5.10 shows the results for the relative network lifetime under different traffic loads with respect to the lifetime of W-MAC. The network lifetime is defined in this work as the time elapsed when the first node dies. We show the relative value of the lifetime so that the results can be appreciated on the same scale for all protocols and all values of the Inter Packet Interval (IPI). However, to get a rough idea of the absolute values when using two AA batteries, the lifetime of W-MAC is 90 days when the IPI is 1 s and 3 years when the IPI is 60 s in mean values.

The Inter Packet Interval (IPI) between the generation of application packets has been varied from 1 s (high traffic) up to 60 s (low traffic) which are typical values in smart cities applications [48]. The bar plot shows the median values for each scenario over the 50 repetitions and the confidence intervals of 95%. The results show the superiority of eLoBaPS over its predecessor LoBaPS (up to 17%) and the reference W-MAC (up to 40%) in all scenarios. Studying the maximum and minimum outcomes for all the repetitions (not shown in the figure), we can affirm that the lifetime improvement can actually go up to 77% in the best case of eLoBaPS compared to the worst case of W-MAC. This is because of the two main features of the protocol: the energy consumption due to packet transmissions is well distributed among all the feasible parents and the nodes that are excessively consuming energy turn off their main radio till they keep up with the energy consumption of their neighbors.

At the same time, it is interesting to remark that in low traffic scenarios almost all the protocols present very small differences in the resulting lifetime. The reason behind this is that as the IPI increases, the impact of the radio communications on the overall energy consumption decreases. Fig. 5.9 supports this fact by depicting the contribution to the overall energy consumption of the different power states of an average node running W-MAC: main MCU in active mode, main MCU in Low Power Mode (LPM), main radio, and wake-up radio. The radio contributions (Main Radio and WuR) comprise transmission, reception and listening for each one. In this figure, it can be noticed that the contribution of the radio modes (Main Radio and WuR) is the most significant to the total only for high traffic scenarios (1 s IPI). On the contrary, in low traffic scenarios, the major contribution to the overall energy is just the LPM, that is, because of the silent power consumption when the node is sleeping.

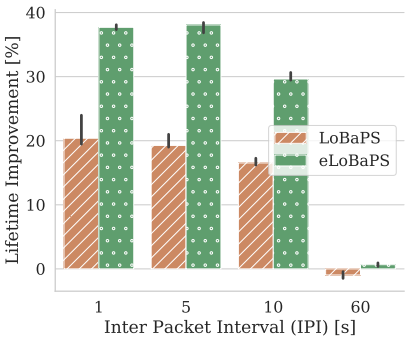


Figure 5.10: Lifetime improvement over W-MAC for different traffic loads

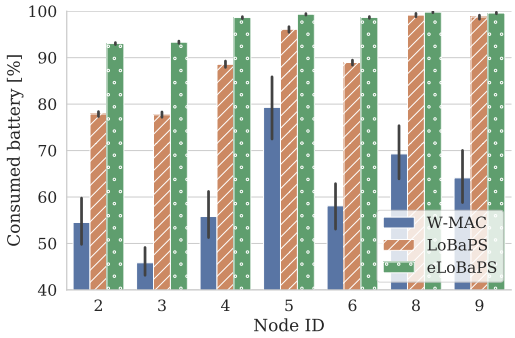


Figure 5.11: Consumed battery of nodes at 1 hop from the sink when the first node dies (higher means a better load balancing)

5.6.4 Battery consumption

The battery consumption of each node at only one hop away from the sink is depicted in Fig. 5.11 at the instant when the first node of the network dies, for a 10 s IPI as an example of the general behavior. The topology of the network imposes constraints to the amount of load balancing that can be achieved with good performance. Leaf nodes (2 hops away from the sink) do not consume the same order of energy than relays, because they do not wake up often to listen to the main radio channel. This is the reason why we do not show their battery consumption in this figure. However, the topology still generates more energy consumption on some nodes (nodes 8 and 9) because they have more chances to be woken up by some child and waste energy listening to the main radio channel. The point of this figure is to analyze how equally distributed is the energy among the network. The goal is to have all nodes consuming approximately the same amount of battery when the first one dies. In the ideal case where the network consumes all the batteries in a balanced way we would expect that all the nodes die at the same time, that is, showing 100 % battery consumption. However, we can see that in W-MAC there are nodes that have only consumed half of their batteries when the first node that dies consumed it all. That amount of remaining battery not used is the reason why the lifetime is shorter because the load is not equally shared. On the contrary, in LoBaPS, the maximum remaining battery for a node at one hop from the sink is of 22%, while in eLoBaps it is of 7%, proving the improvement of the load balancing algorithm in terms of energy efficiency.

5.6.5 Control overhead

We also consider a metric to measure the number of control packets transmitted on the whole network. This metric is calculated with Eq. 5.5.

$$c = 100 \frac{\#network\ control\ packets}{\#\ app\ packets\ at\ the\ sink} \tag{5.5}$$

The results are shown in Fig. 5.12 evidencing a low control overhead for both

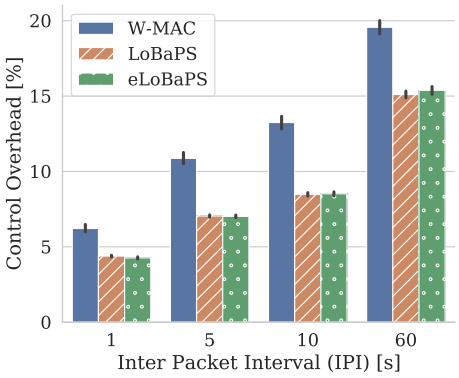


Figure 5.12: Control Overhead

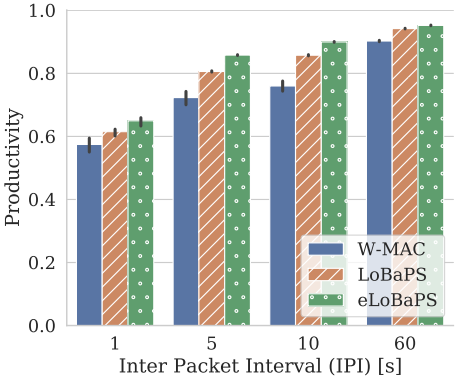


Figure 5.13: Productivity for different IPIs

LoBaPS and eLoBaPS compared to W-MAC. The main cause of this is that in W-MAC, whenever a backup parent dies, its children need to repair the routing structure by generating new control packets. In contrast, in both LoBaPS and eLoBaPS, this is not necessary since the initial routing structure can still be used as long as there is still connectivity in the network graph.

5.6.6 Productivity

Naturally, the overall application data received at the sink is different for each traffic scenario, thus making somehow doubtable the comparisons for both the PDR and the lifetime. For this, we have come up with a productivity formula that divides the number of application packets correctly received at the sink by the simulation elapsed time, so-called *simulation disconnectivity time* because the network stops being a connected graph. In addition, it is scaled by the scenario parameters (IPI and network size) so that the value can be compared between all the simulations.

$$Productivity = \frac{\#\, app\, packets\, at\, the\, sink}{disconnectivity\, time} \frac{IPI}{\#\, nodes} \tag{5.6}$$

This way we conceive a metric of the productivity of the network that somehow combines the notion of latency, reliability and lifetime in a single value: productivity. Clearly, we want this value to be as higher as possible. We can also see this formula in an equivalent way, as the ratio of the expected disconnectivity time (calculated based on the number of application packets received at the sink, the IPI and the number of nodes) and the simulation disconnectivity time:

$$Productivity = \frac{expected\, disconnectivity\, time}{simulation\, disconnectivity\, time} \tag{5.7}$$

Notice that the maximum in mean values is 1 since the simulation disconnectivity time cannot be shorter than the expected one since the nodes can not generate packets faster than the IPI in mean values. So smaller values of productivity mean that it takes more time than expected to deliver the packets to the sink.

The results of this metric are illustrated in Fig. 5.13, where we can see a trend of improvements between W-MAC, LoBaPS, and eLoBaPS, for all the traffic loads.

The reason for this is linked to the reduced number of retransmissions that are required in eLoBaPS as well as its shorter decline slope (described in Section 5.6.1) which turns into a smaller disconnectivity time. This means that eLoBaPS is more productive because it gets the job done in a shorter period of time. Coupled with this result, we can see that the productivity increases as the IPI increases too for all protocols. The reason is that with low IPI there are more collisions and so fewer packets are successfully delivered to the sink.

5.7 Conclusions

This chapter introduces LoBaPS a load balancing parent selection algorithm. The main idea is to allow all feasible successors to compete for a packet forwarding when a node transmits a packet, taking advantage of the always-on feature of the Wake-Up Radio. We showed that it overcomes the single point of failure problem at the preferred parent of traditional RPL with Objective Function Zero and MinHop metric. Also, we introduced eLoBaPS, an improvement of LoBaPS towards the ideal load balancing. The main idea is to prioritize feasible successors with more remaining battery during the competition for data forwarding. At the same time, it mitigates the main radio listening energy wastage by turning off the most consuming nodes until they keep up with the energy consumption of their neighbors. Moreover, the same idea can be applied with a different metric instead of the energy in order to optimize a different parameter of the network, for instance, the packet queue size. By improving the load balancing towards the ideal case, it extends the network lifetime up to 77%. In addition, the network behavior becomes more stable over its lifetime and the decline with degraded performance is shorter.

An important point that can be concluded based on the results with varying IPI, is that with WuR it is not necessary to make an effort on the design of the protocol for low traffic scenarios (high IPI). In those scenarios, the radio communication power is not significant. In contrast, research efforts should focus on high traffic scenarios (that is, with an IPI of 10 s or less as suggested by our results), thus emphasizing the importance of the CCA function already discussed in our past and present works.

In this chapter, we investigated how WuR can impact the performance of a whole network. For this, we used an existing routing structure, built by RPL, and we showed that our solution achieves good performance. However, the assumption that WuR and the main radio have the same range is not realistic by the current advance in WuR. Also, reducing the range of the main radio increases the number of intermediate hops to deliver a data packet. In the next chapter, we propose a new protocol stack that benefits from the characteristics of WuR.

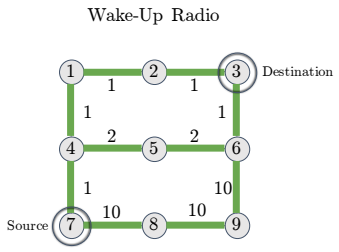
Chapter 6

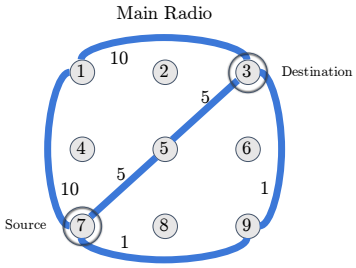
REFLOOD: Reactive routing protocol

Contents

6.1 Introduction

One of the main drawbacks of WuR technology is its very short range. This creates a range mismatch with the main radio, which has a longer range due to better sensitivity. Consequently, the use of WuR leads to very dense networks, because the nodes must be close to each other, to communicate through the very short range of WuR. From a routing layer perspective, the network can be considered as a graph, where the nodes are the vertices and the links between nodes are the





Path name	Nodes in the path	Cost
[*] WuR Path A	7, 4, 1, 2, 3	1 + 1 + 1 + 1 = **4**
WuR Path B	7, 4, 5, 6, 3	1 + 2 + 2 + 1 = 6
WuR Path C	7, 8, 9, 6, 3	10 + 10 + 10 + 1 = 31

Path name	Nodes in the path	Cost
MR Path A	7, 1, 3	10 + 10 = 20
MR Path B	7, 5, 3	5 + 5 = 10
[*] MR Path C	7, 9, 3	1 + 1 = **2**

[*]: Shortest path

Path name	Combination	Cost
Path A	WuR Path A + MR Path A	4 + 20 = 24
[*] Path B	WuR Path B + MR Path B	6 + 10 = **16**
Path C	WuR Path C + MR Path C	31 + 2 = 33

Figure 6.1: Dual graphs. MR: Main Radio

edges. A cost might be associated with each edge, related to some communication performance metric of that link, such as the *expected transmission count* (ETX). In WuR networks, there are two radios, which translates into two sets of links. As a result, we can consider a multigraph with two sets of edges in parallel [49], one for the WuR links, and the other for the main radio links. The cost associated with a link on one set of edges is independent of the cost on the other set because the communication channels are different. A priori, they use different modulation techniques, and may even work in different frequency bands. However, the links on the set of edges of the main radio are only activated when a message is transmitted on the appropriate WuR links so that both nodes on the main radio link are awake and ready to communicate. In other words, the difference with a classical multigraph is that the availability of main radio links is dependent on the activity of the WuR links, which evolves with time. This means that, although the costs of the two types of links are independent, both sets of edges need to be considered simultaneously to search for the best paths. The shortest path from a source node to a given destination is a combination of a path on the set of edges of the main radio and a path on the set of edges of WuR. Notice that these partial paths, may not necessarily be the shortest individually in each set of edges.

An example of this is shown in Fig. 6.1, where we assume that only the green and blue links are available and that the source is node 7, and the destination is node 3. Notice that the best combination of paths is *Path B* because it minimizes the total cost. However, *WuR path B* is not the shortest path on the set of edges of WuR, and *MR Path B* is not the shortest path on the set of edges of the main radio. So the final shortest path depends on the best combination of WuR path and main radio path that minimizes the total cost associated. Finding such a path is a difficult task in WSN applications because it requires that all the nodes exchange link-state information in the network. Typically, in WSN, we try to avoid such complexity in the communication protocols because the end-devices are resource-

constrained. Conversely, if this challenge is not addressed, the data packet would go through every node that forms a path from a source to destination, even though the source and the destination are in the same range of the main radio, canceling out the benefits of the technology. This problem is crucial to spread the wake-up radio technology, and it has not been studied extensively yet.

In this chapter, we propose a first step towards an efficient solution to route the wake-up signal in such networks, that optimizes the use of resources.

The contributions of this chapter are:

Contribution

- We show that WuR is more suited for low-traffic scenarios because the low data rate increases the time over the air required to send a signal and the receiver is very sensitive to collisions and interferences.
- We found that reactive routing is a better approach to relay the wake-up signal to wake up a destination because this technology is very sensitive to errors or collision on the WuR channel that can make the routing structure very unstable in time, as we showed in Chapter 3 [21]. With that in mind, we propose in this chapter REFLOOD, a reactive routing protocol, addressing the range mismatch problem. We evaluate its performance by comparing it to a proactive routing strategy through an exhaustive series of simulations in ContikiOS/COOJA [50].

Publication: S. L. Sampayo, J. Montavont and T. Noel. PROPL and REFLOOD: Proactive and Reactive Protocols for Wake-Up Radio Routing in IoT, submitted to Elsevier Ad Hoc Networks (pending review), CiteScore 7.8, Impact Factor 3.643.

6.2 Background on routing protocols for WSN

In computer networks, routes can be calculated with a proactive, reactive, or hybrid approach. In proactive protocols, the nodes build a routing structure as soon as they are initialized, even if there is no data to send. Then, the nodes should maintain their routing structure throughout the network lifetime. The Routing Protocol for Low Power and Lossy Networks (RPL) [12] is the Internet Engineering Task Force (IETF) standard for multi-hop routing in WSN. RPL is a proactive protocol that builds a Destination-Oriented Directed Acyclic Graph (DODAG) based on distance vectors, and each node selects a preferred parent when joining the network. However, RPL still presents some open problems, such as inefficient parent selection and instability when the Expected Transmission Count metric is used [47].

On the other hand, in *reactive* protocols, the nodes only search for paths as soon as a data packet is generated. LOADng [51] is an example of this approach, where

the nodes request routes by flooding the network. Then, the nodes keep a routing table with the recent destinations for a certain amount of time. However, the main concern about reactive protocols is the path discovery process for new or broken routes, which could potentially incur additional delay, additional control signaling, and data retransmissions [51].

The recent works that focus on routing protocols for low power and lossy networks have discussed the benefits and drawbacks of RPL, as an exponent of the proactive approaches, and LOADng, representing the reactive approaches. The authors in [52] compared RPL with LOADng and showed that the latter one is especially better for point-to-point (one-to-one) and point-to-multi-point (one-to-many) traffic patterns. The following year, the same authors presented in [53] an extension called LOADng-CTP, targeting multi-point-to-point (many-to-one) traffic, and was compared to RPL. The resulting performance of both protocols is very similar, only that the reactive approach presents better control overhead.

Tripathi et al. have shown in [54] that RPL has lower overhead and lower delay, whereas in LOADng the control overhead is proportional to the network size. Also, the reactive protocol has higher buffer size requirements compared to RPL in the non-storing mode because of control packet flooding and buffering of data packets. That is not the case when comparing LOADng and RPL in storing mode, where the RAM occupancy is very similar in both protocols. However, those results are dependent on the traffic pattern and the topology in use. On the other hand, the reactive protocol's complexity is lower and it proves to be very effective in low traffic scenarios.

Newman et al. explain in [55] that the proactive approach is generally the choice when the nodes do not have energy constraints. Moreover, the authors also affirm that proactive protocols are more efficient in terms of delay when the traffic load is high.

Recently, Sobral et al. explained in [56] that RPL presents severe limitations and drawbacks, such as the weak support of mobility and P2P traffic, restrictions for multicast transmissions, and poor adaption for dynamic throughput. Many solutions and extensions have emerged in the past few years to mitigate those issues, but the authors identify open problems that remain. On the other hand, they conclude that LOADng is a potential solution but has been less studied than RPL.

To conclude, the debate between proactive and reactive approaches is still open in WSN and the solution typically depends on the application constraints. Nonetheless, there seems to be a point in favor of using reactive strategies in low traffic or delay-tolerant scenarios.

WuR is a new technology that has emerged in the past few years and is especially interesting for low traffic and asynchronous applications [57]. Unfortunately, a proactive approach, such as RPL, presents issues when a WuR-based protocol is used at the MAC layer because the DODAG that is built is not optimal since it is limited by the very-short-range links of the WuR topology. Based on this analysis, we propose a reactive routing protocol. Spreading WuS across neighbors allows a source to explore multiple paths towards a destination. Such a strategy will be beneficial against interferences and collisions in the WuR medium. Besides, routes will be created on-demand regarding the current state of the network. In low traffic scenarios, such an approach will be scalable and seems more effective than maintain-

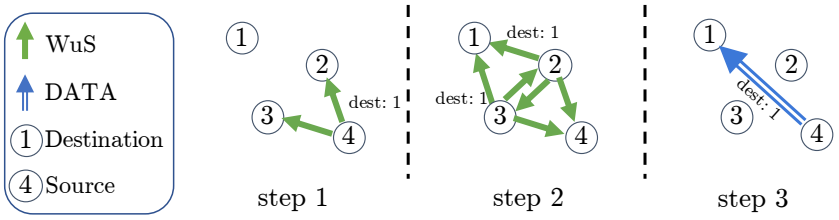


Figure 6.2: Example of data communication in REFLOOD. Node 4 is the source and node 1 is the destination.

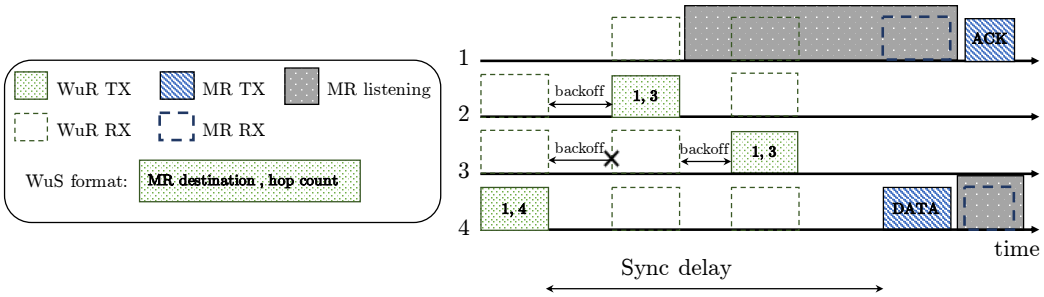


Figure 6.3: Reactive protocol timeline (MR: Main radio).

ing routes during the whole network lifetime. In the following section, we present REFLOOD, a reactive routing protocol using WuR. To evaluate its performance, we present a proactive WuR-based routing protocol, PROPL, in the subsequent section.

6.3 REFLOOD - Reactive protocol

In this section, we present REFLOOD, a WuR-based routing protocol whose name combines the words *reactive* and *flood*. The idea behind REFLOOD is to flood the network with the address of the destination via the WuR channel. Eventually, the flood will reach the destination and this one will wake up its main radio to communicate. This way, we take advantage of multiple paths to reach the destination and increase the chances of waking it up correctly. As a result, the impact of network dynamics due to collisions and interferences on the communication channel should be mitigated.

For example, in Fig. 6.2, the source initiates the flood by transmitting a WuS with the address of the destination. The WuR is always-on, so all the WuR neighbors will receive this WuS. Every WuR neighbor that receives this WuS will forward it only once, except for the destination. When the latter one receives the WuS it wakes up its main radio and waits for the data frame. Finally, the source transmits the data frame on the main radio channel. A timeline for this process is shown in Fig. 6.3. We showed in Chapter 5 that a clear-channel assessment function is required to limit collisions when transmitting the WuS. Such a mechanism is included in this work.

6.3.1 Flooding challenges

The classic problem of a flooding protocol is that the flood may extend to the whole network and that the nodes may retransmit the same frames infinitely. REFLOOD limits such extension by adding a hop-count in the WuS. Also, to avoid forwarding duplicates infinitely, each node starts a short-duration timer after forwarding a WuS. During that timer, all the following WuS received are not forwarded.

6.3.2 WuS size

WuR uses a low data rate which translates into a long time over the air occupying the medium. For that reason, the size of the WuS message must be as small as possible. The WuS requires one field for the address of the destination. Typically, in the WuR literature, an address is described with 8 bits, so we can address up to 255 nodes in a flat address scheme. We set the hop-count field to 2 bits. Such value allows building routes including up to 5 nodes (including source and destination). We showed in [21] that the performance of WuR-based MAC protocols seriously degrades when the source requires more than 5 hops of the WuS to reach the destination. That makes a total of 10 bits in this protocol.

6.3.3 Algorithm

REFLOOD operates according to the pseudocode shown in Algorithm 1. The transmission process waits until it is signaled by upper layers. In the case of a source node, such a signal is a trigger for data transmission. First, if there is a flood in progress, the WuR module is blocked, and the process waits until it becomes available. Then, the frame for the WuS is prepared for the WuS transmission with the address of the destination and the maximum extension of the flood, given by $hops_{max}$. In our implementation, the value for this parameter is 4, since we use 2 bits for the hop-count field, for the reasons explained in Section 6.3.2, based on [21]. Next, the flood is started by transmitting the WuS. Once the flood is done successfully, the data frame is sent on the main radio. Finally, the upper layers of the communication stack are notified of the errors or the success of the transmission.

The pseudocode used for the flooding process is shown in Algorithm 2. Such waiting is limited in time, and it can fail if the channel stays busy at the end. In such a case, the flood returns a failure state. Otherwise, the WuS is transmitted and the WuR module is blocked for a period given by the parameter Δt_{sync_delay}, corresponding to the sync delay defined in Section 2.4 (see also Fig. 6.3). This period is necessary for the WuS to propagate towards the destination and wake it up. The WuR module must be blocked for two reasons. First, to avoid infinite flooding, the source node must ignore the reception of the duplicated WuS issued from the current flood process. Second, to avoid creating new flood processes, triggered by other concurrent requests to transmit data from upper layers, that could generate collisions in the medium.

The pseudocode used to block the WuR module is shown in Algorithm 3. The function Block_WuR(delay, event) takes two arguments as inputs: the interval of time during which the WuR will remain blocked, and an optional event that can stop the wait prematurely. In the procedure, the function disables the WuR module

Algorithm 1: WuR_send_process(*data*)

Input: *data* frame

```
1  while true do
2      Wait for data transmission request from upper layers;
3      /* Case of source node                                   */
4      /* Check if there is a flood in progress                 */
5      if WuR is blocked then
6          wait until it is released
7      /* Prepare WuS frame                                     */
8      WuS.destination ← SINK_ADDRESS;
9      WuS.hop_count ← hops_max − 1;
10     if Flood(WuS) = success then
11         Send data frame on the main radio
12     Notify transmission status to upper layers;
```

Algorithm 2: Flood(WuS)

Input: WuS frame

Output: The flood status (success or failure)

```
1  Wait_for_WuR_CCA();
2  if WuR channel is busy then
3      Return failure;
4  Transmit WuS on WuR;
5  Block_WuR(Δt_sync_delay, NULL);
6  Return success;
```

Algorithm 3: Block_WuR(*delay*, *event*)

Input: Interval of time *delay* during which the WuR remains blocked, or *event* that can release it

```
1 Disable WuS reception;
2 Set timer to delay;
3 Wait for the timer to expire or event;
4 Enable WuS reception;
5 Notify WuR released;
```

Algorithm 4: Wait_for_WuR_CCA()

```
1 i ← N_cca;
2 do
3 |   i ← i − 1;
4 |   backoff ← Δt_WuS/2 + random(0, Δt_WuS);
5 |   Set timer to backoff;
6 |   Wait for timer to expire;
7 while WuR channel is busy and i > 0;
```

interrupts to ignore the reception of WuS. Then, a timer is used to wait for the specified interval. Finally, once the timer expires or the optional event is issued, the WuR interrupts are re-enabled, and a signal is generated. Such signal notifies any potential process which may be waiting for the WuR module to acquire it, particularly WuR_send_process(), described previously in Algorithm 1.

Algorithm 4 shows the pseudocode used to perform a clear channel assessment on the WuR channel, based on the contributions of Chapter 4. Essentially, the function waits for a short random backoff of approximately Δt_{WuS}, which is the length of the WuS in time. If the channel is still busy after such delay, another random backoff is drawn and the channel is checked again for a maximum of N_{cca} times, which is set to 10 in our implementation, based on Chapter 4.

Finally, the reception process of the WuR module is described by the pseudocode in Algorithm 5. This process simply waits for activity on the receiver of the WuR module. When a WuS is received, the destination field is read and compared to the address of the node running this process. If there is a match, it is the case of a destination node. Then, it turns on the main radio and waits for the data frame, while blocking the WuR module to avoid the aforementioned problems (infinite flooding and creating new flood processes). Once a period given by Δt_{sync_delay} has passed or the data frame has been received, the main radio is turned off and the process continues. On the other hand, if the destination field address does not match the address of the node, the hop count field in the WuS is checked. If such field is zero, the WuS is dropped and the process continues, to limit the extension of the flood. Otherwise, the node decrements such field and forwards the WuS, contributing to the flood.

Algorithm 5: WuR_receive_process()

```
1  while true do
2      Wait for WuS reception;
3      if WuS.destination = me.address then
4          /* Case of destination node                          */
5          Turn main radio on;
6          Block_WuR(Δt_sync_delay, data reception event);
7          Turn main radio off;
8          if data reception OK then
9              Notify data to upper layers;
10     else if WuS.hop_count = 0 then
11         /* Hop limit reached, drop WuS                       */
12         continue;
13     else
14         /* Case of WuS relay node                            */
15         WuS.hop_count ← WuS.hop_count - 1;
16         Flood(WuS);
```

6.3.4 Sync delay determination

We defined the *sync delay* in Section 2.4 as the elapsed time between step 1 and step 3 in Figure 2.7, that is, between the transmission of the WuS by the source, and that of the data frame on the main radio. The value for the sync delay in the algorithm of REFLOOD is fixed by Δt_{sync_delay}. If this value is too short, the source would send the data frame before the destination receives a copy of the WuS to wake up, so the data frame would be lost. On the contrary, if the value is too long, the destination would spend a long time in listening mode on the main radio in vain wasting energy consumption and increasing the latency of the communication. An analysis of such scenarios was presented in Chapter 3. There, it is concluded that a medium sync delay appears to be a reasonable compromise between latency and power consumption for scenarios with significant interferences and collision probability.

In REFLOOD, the sync delay has to let the WuS traverse some path from source to a destination within the flooding process. That means, that in the case of the sources that are further away from the destination, the sync delay must be long enough to contain a sequence of $hops_{max}$ WuS transmissions with an associated CCA procedure for each one. Therefore, to fix the value of Δt_{sync_delay} in our implementation, we take into consideration the maximum number of hops that the WuS can propagate ($hops_{max}$), the mean duration of the CCA backoff ($\bar{\Delta} t_{cca}$), the duration of each WuS transmission (Δt_{WuS}) and the time required to process it and forward it (Δt_{proc}). The formula to compute it is given by Equation 6.1

$$\Delta t_{sync\_delay} = hops_{max}(\bar{\Delta} t_{cca} + \Delta t_{WuS} + \Delta t_{proc}) \tag{6.1}$$

Moreover, according to Algorithm 4, the mean duration of a single CCA backoff

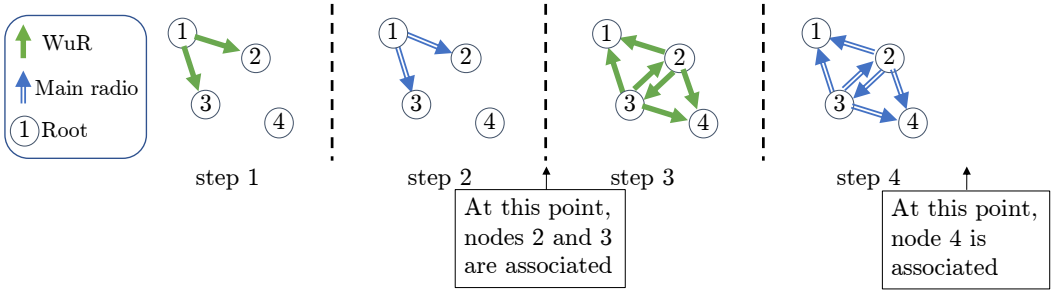


Figure 6.4: Building an RPL DODAG with reduced main radio transmission power.

is equal to the length of the WuS in time, which means that $\bar{\Delta}t_{cca} = \Delta t_{WuS}$. Then, Equation 6.1 can be simplified as in Equation 6.2

$$\Delta t_{sync\_delay} = hops_{max}(2\Delta t_{WuS} + \Delta t_{proc}) \tag{6.2}$$

Notice that the length of the WuS in time is given by the data rate (D) and the size of the WuS frame (L), as of Equation 6.3.

$$\Delta t_{WuS} = \frac{L}{D} \tag{6.3}$$

6.4 Simulation framework

In agreement with the previous chapters, we used the same simulation framework based on WaCo [17], a COOJA extension, reproducing the actual firmware that runs on real devices.

6.4.1 Protocols

We implemented REFLOOD according to Section 6.3. To compare with the traditional solution in WSN, we added an implementation of a duty-cycled MAC protocol that does not use the WuR. In this case, there is no routing required because all the nodes are 1-hop neighbors on the main radio. We used ContikiMAC [38] with a channel check rate of 8 Hz and phase synchronization.

Besides, we implemented PROPL, which is a WuR-based protocol whose name combines the words *proactive* and *RPL*. RPL is the IETF proactive routing protocol standard, so its usage for comparison with REFLOOD seems relevant. RPL can be seamlessly used over a WuR-based MAC protocol if we limit the range of the main radio to match that of WuR for all RPL control packet transmissions. Then, notice that the DODAG is built based on the WuR topology which is formed by the short-range links. For this reason, in PROPL, we reduce the main radio transmission power whenever an RPL control packet is to be sent on the main radio. Also, we cannot use WuR to send RPL control packets because we need to limit the size of the WuS to as small as possible.

Fig. 6.4 shows an example of the process to build the routing structure in PROPL. The root (in this case, node 1) initiates the process by sending a WuS to wake up the main radio of its WuR neighbors (nodes 2 and 3, step 1).

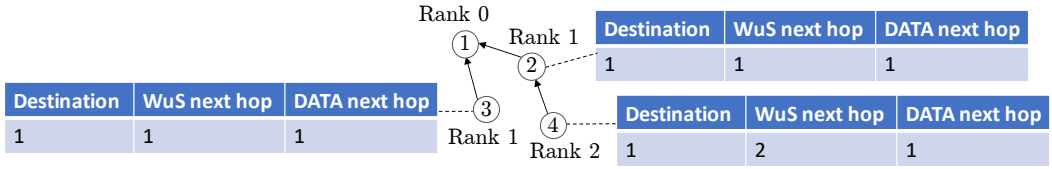


Figure 6.5: Resulting DODAG.

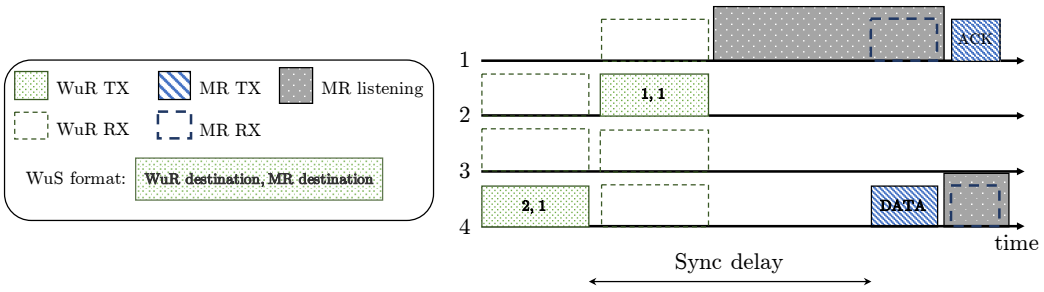


Figure 6.6: Proactive protocol timeline (MR: Main radio).

Then it sends a DIO to its neighborhood on the main radio (step 2) which allows nodes 2 and 3 to associate to the RPL DODAG and compute their ranks. Notice that the control packets are transmitted on the main radio but with reduced transmission power so that the range matches that of the WuR. Then, these nodes will generate their own DIOs and send them to their WuR neighbors using the same process (first sending a WuS to their WuR neighbors in step 3 and then transmitting the DIO on the main radio in step 4). The resulting DODAG is depicted in Fig. 6.5, which shows the extending routing tables associated with each node.

When node 4 has a data packet to send, it uses this DODAG to reach node 1 in the same way as in the example of Fig. 2.7. A timeline of this process is shown in Fig. 6.6. First, the source sends a WuS to node 2. This WuS is also received by node 3 but ignored because it does not match the WuR destination included in the WuS. Then, node 2 keeps its main radio in sleep and forwards the WuS to node 1. Upon reception, node 1 wakes up its main radio and waits for the data frame. Finally, after the sync delay expires, node 4 transmits the data frame on the main radio channel. In general, this frame is followed by an acknowledgment from the destination to the source to confirm the reception. In PROPL, the value of the sync delay is computed with Equation 6.2, in the same way as in REFLOOD.

In this protocol, the WuS requires two fields: one for the address of the WuR next-hop neighbor which relays the WuS to the destination, and another one for the address of the final destination of the WuS, i.e. the destination node of the pending data packet. In other words, the WuS needs a piece of information about the routing of the WuS, and another piece of information about the destination of the data packet. Since REFLOOD uses 8 bits to address a node, the WuS size is 16 bits for the proactive case.

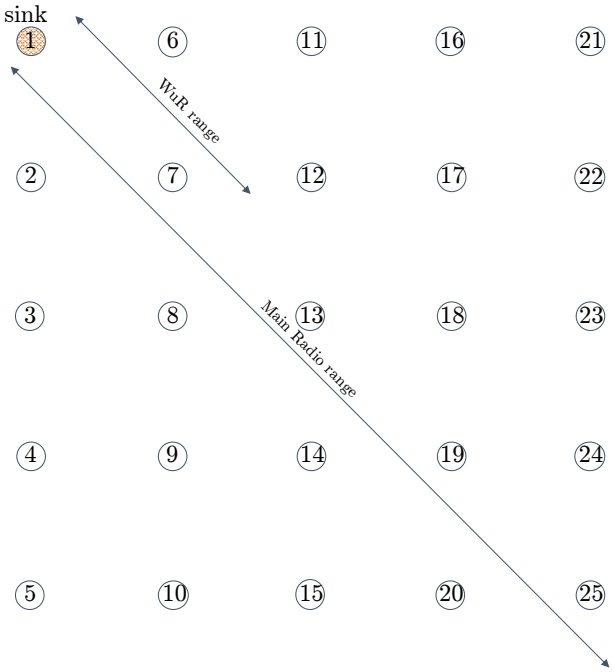


Figure 6.7: Simulated grid topology

6.4.2 Simulation setup

The protocols were evaluated in a controlled grid topology as well as in a random one, as depicted in Fig. 6.7 and Fig. 6.8 respectively. First, we show the main results with the grid topology and then explain the differences that appear with the random one. In both cases, the node with id 1 is the only destination (sink) in the scenario, and all the other 24 nodes are sources. In the case of the random topology, all the source nodes were placed randomly in a square of 40 x 40 m, in such a way that the network is a connected graph concerning the WuR links. The WuR and main radio ranges are depicted in both figures. For example, in Fig. 6.8, the WuR neighbors of node 1 are nodes 2, 6, 12, 13, 20, and 22. On the contrary, nodes 5, 18, and 24 are not in the WuR range of node 1. On the other hand, all the nodes in both topologies are 1-hop neighbors on the main radio, when used at full power. For example, in Fig. 6.8, nodes 1 and 16 are main-radio neighbors.

The source nodes generate packets with a random process that is illustrated in Fig. 6.9. A timer with a fixed period T is used to generate events. When an event is triggered, a uniform random variable u of range $(0, T)$ is drawn and the node uses this value to wait for a delay before generating the packet. The moment when the packet is generated is shown with a cross. All the simulation last long enough so that each node generates 30 packets at the application layer.

We vary the value of T to generate low, medium, and high traffic scenarios. We also vary the values of the WuR data rate. Moreover, we use the Unit Disk Graph Model (UDGM), a medium model included in COOJA, with a packet reception success ratio of 90 % for both the WuR channel and the main radio channel, according to the results in [58]. We perform simulations for all the possible combinations

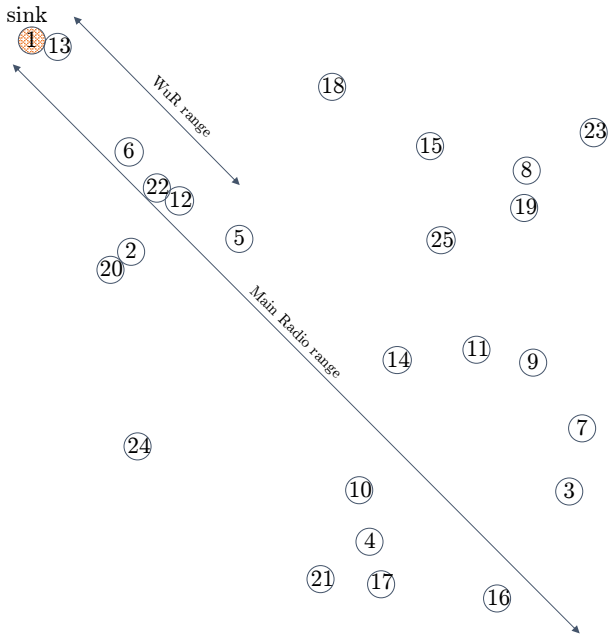


Figure 6.8: Simulated random topology

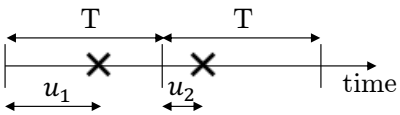


Figure 6.9: Example of the packet generation process

Table 6.1: Simulation parameters

Parameter	**Value**
Packet generation period (T)	10 s, 60 s, 1000 s
WuR data rate (D)	1 kbps, 10 kbps
Medium model	UDGM
RX success ratio	90%
MAC layer	CSMA (Contiki version)
CSMA minBE	3
CSMA maxBE	5
CSMA maxBackoff	4
CSMA maxRetries	3
ContikiMAC channel check period	125 ms
Main node	Sky mote [36]
WuR HW prototype	[16], [44], [58]
WuR Supply Voltage	3 V
WuR TX current	14.3 mA
WuR RX current	0.4 mA
WuR idle listening current	7.6 μA
WuS size (L)	10, 16 bits
$hops_{max}$	4
N_{cca}	10
Δt_{proc}	1 ms

between those parameters and repeated the experiment 48 times for every single combination, resulting in a total number of 1728 simulations. The confidence intervals of 95% ensure that our measurements are statistically significant. A summary of the simulation parameters is given in Table 6.1.

As a consequence of such configuration of parameters, the calculation of the sync delay for each protocol using Equation 6.2 is given in Table 6.2. Notice that the sync delay can be compared to the time that takes a source to wake up a destination in ContikiMAC. In such protocol, that time goes from zero, when the destination phase is well synchronized with the source, up to the channel check period, which is 125 ms by default.

In our application, we use Carrier Sense Multiple Access (CSMA) for each data transmission. When a packet is generated at the application layer (indicated by a cross in Fig. 6.9), the CSMA mechanism starts operating by waiting for a random delay to avoid potential collisions. Once such delay is done, a data transmission request is signaled to REFLOOD, as in Algorithm 1. Typically, the unit backoff period of the CSMA implementation included in ContikiOS corresponds to the parameter *aUnitBackoffPeriod* of the IEEE standard 802.15.4 [59]. Its value is proportional to the channel check interval of the underlying radio duty cycle. In the case of ContikiMAC, such an interval is 125 ms. Analogously, in WuR-based protocols, we set the value of the unit backoff period of the CSMA to the sync delay.

Table 6.2: Sync delay calculation

Protocol	Data rate (D) 1 kbps	10 kbps
REFLOOD	84 ms	12 ms
PROPL	132 ms	16.8 ms
ContikiMAC	[0 - 125] ms	

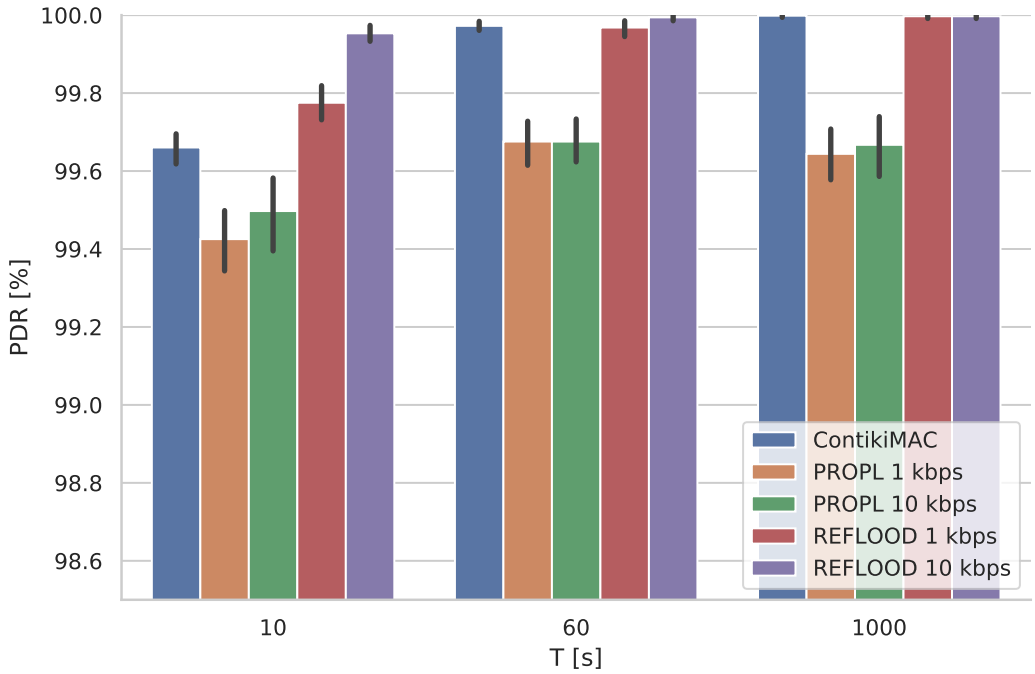


Figure 6.10: Packet Delivery Ratio

6.5 Results

6.5.1 PDR

The Packet Delivery Ratio (PDR) is computed by counting the number of packets received at the sink and dividing it by the total number of packets generated at the application layer of all nodes. That means that it does not take into account potential duplications generated by the CSMA mechanism. Fig. 6.10 shows a bar plot with the mean values and 95% confidence intervals ensuring that our measurements are statistically significant, for the case of the grid topology of Fig. 6.7.

We can see in Fig. 6.10 that REFLOOD presents the highest PDR, similar to ContikiMAC. On the contrary, PROPL presents the lowest one. As the low data rate is one of the drawbacks of WuR, increasing it results in an improvement of the PDR.

When a packet is lost, the CSMA layer triggers a new WuS on the main sender,

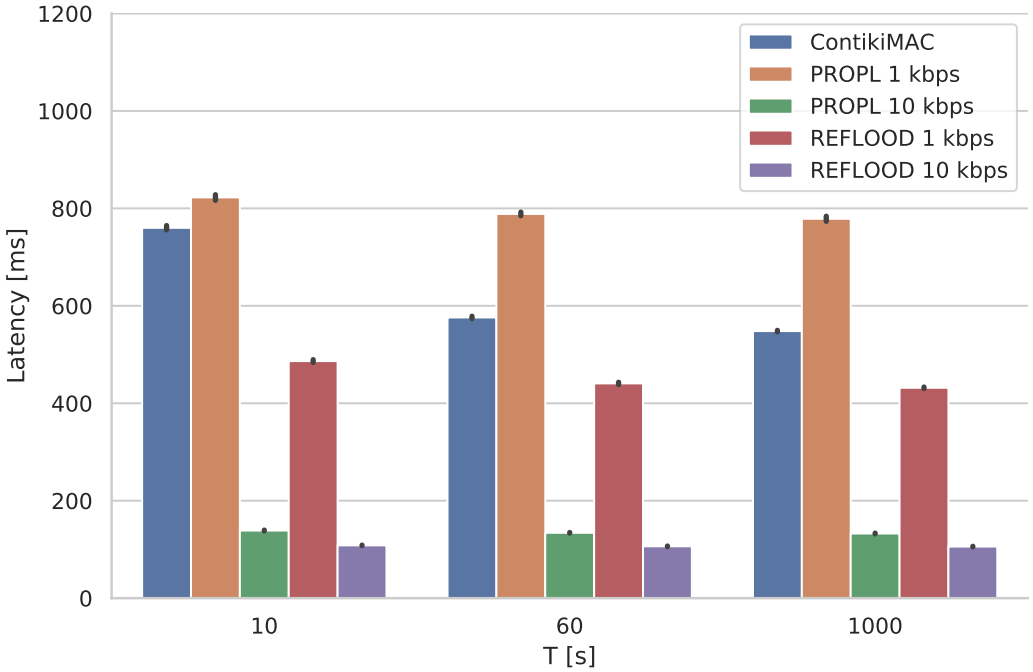


Figure 6.11: Latency

restarting the communication process to reach the destination and wake it up. In PROPL, if only one node on the WuS path fails, the whole CSMA attempt fails, and triggers a new retransmission. In the simulated topology, the longest WuS path includes 5 nodes, so as many possibilities to fail the transmission of the WuS. We can secure the WuS with acknowledgements, but it has been shown previously in [21] that using WuS ACKs is counterproductive because the power consumption and latency are increased and the PDR even more reduced because the channel is very sensitive to interferences and collisions. Then, the only acknowledgement that we used is the one following a data frame on the main radio. On the other hand, in REFLOOD, when an intermediate relay fails, there are some chances that another feasible forwarder has succeeded in relaying the WuS. That means that for an equal number of maximum CSMA attempts, REFLOOD has more chances to deliver the packet successfully, thus a higher PDR, thanks to multiple paths. In conclusion, the flooding introduced by REFLOOD allows the WuS to reach the destination via multiple paths, increasing the chances to wake it up and deliver the data successfully.

6.5.2 Latency

The latency is computed as the end-to-end delay at the application layer. We count the elapsed time between the instant at which the packet is generated in such layer and the point at which the corresponding acknowledgment is received back. This means that the CSMA backoffs and the time elapsed between consecutive retransmissions are included in the end-to-end delay value. Also, we only account for successful deliveries at the application layer to compute it.

Fig. 6.11 shows a bar plot with the mean values and 95% confidence intervals.

The results show that REFLOOD at a WuR data rate of 10 kbps presents the shortest latency of all protocols. Furthermore, its mean value is not affected by the traffic load. The main reason is that the length of the WuS path in time is shorter than that of other protocols and the required time to connect to a destination in ContikiMAC. Fig. 6.11 shows that for all traffic loads, the order from lower to higher latency is as follows: REFLOOD 10 kbps, PROPL 10 kbps, REFLOOD 1 kbps, ContikiMAC, and PROPL 1 kbps. Notice that this is exactly the order of increasing values of sync delay in Table 6.2. This make sense, since the resulting CSMA delay is proportional to the unit backoff period parameter. In our implementation, its value equals the sync delay, as described in Section 6.4.

Furthermore, since the minimum backoff exponential is 3, the resulting CSMA delay of the first transmission attempt is a random number between 0 and 7 times $(2^3 - 1)$ the unit backoff period parameter. This explains the absolute values of latency for each protocol. Moreover, the high PDR results of Fig. 6.10, suggest that REFLOOD requires less retransmissions, which in turn results in a lower end-to-end delay.

In conclusion, the reactive protocol keeps the CSMA retransmissions number low by using multiple paths without consuming more energy on the main radio. This results in low latency and high PDR at the same time for this protocol. Furthermore, it participates to distribute more efficiently the load over the nodes as presented in the next section.

6.5.3 Load balancing

In an attempt to characterize the load balancing of each protocol, we use *Powertrace* [39] and *WurPowerTracker* [17] to monitor the power consumption of all nodes. Then, we make a bar plot for a single run of the simulation to take an example. In this case, the protocol is successful when all the bars are approximately the same high, which means that the load is well shared among the nodes in the network. How a protocol balances the load across the network plays an important role in the performance metrics. The goal is to reduce the maximum value of the power consumption while also distributing the load evenly among the nodes in the network to avoid the funneling effect and extend its lifetime. Fig. 6.12 shows the power consumption per node in ContikiMAC, PROPL, and REFLOOD, for a T of 1000 s and WuR data rate of 1 kbps with RX success rate of 90%.

ContikiMAC, in Fig. 6.12a, is well balanced because all the nodes consume approximately the same: each node is only responsible to transmit its own traffic as there is no multi-hop in this configuration. However, the absolute value of the power consumption of the nodes is quite high compared to that of WuR protocols. Such high values are mainly due to idle listening and overhearing that asynchronous MAC protocols generally suffer from. On the other hand, PROPL, in Fig. 6.12b does not manage to distribute the load evenly among the nodes. The sink is overcharged because of the long idle listening on the main radio. Its power consumption reaches 0.4 mW but it is not shown in the figure for legibility purposes. If we focus on the rest of the nodes, we notice that some of them are especially more charged than others. The reason is simply that those nodes were chosen as parents for many children, and thus forward more packets than their siblings. Notice that these nodes are the ones

that are located close to the sink and around the center of the topology (nodes ids: 8, 12, 13, 14, 18 in the grid topology, Fig. 6.7), typically victims of the funneling effect. Then, REFLOOD, in Fig. 6.12c distributes better the load among the nodes which improves the network lifetime, but the sink is also overcharged (and its power consumption also reaches around 0.4 mW) for the same reason as the proactive case. This observation should be taken into consideration when extending this work to a larger network with multiple hops of the data frame on the main radio. If we increase the WuR data rate to 10 kbps (Fig. 6.13), REFLOOD reduces, even more, the overall energy consumption, because the contribution of the WuR to the overall power consumption is less significant.

6.5.4 Network lifetime

To compute the network lifetime, we take the node with the highest mean power consumption (without taking the sink into account) and used the formula in [42] for a linear battery model. The formula is slightly modified to use the power instead of the current and giving the result in *days* instead of *hours*:

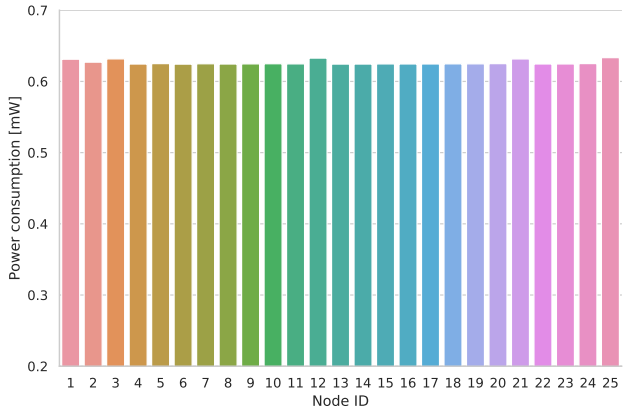
$$T_{life} = \frac{C_{bat} V}{24 P} \tag{6.4}$$

where T_{life} is the lifetime in days, P is the highest mean power consumption among all nodes in milli-watts, V is the voltage of the device in volts, and C_{bat} is the capacity of a single AA battery in milli-amperes per hour ($\approx$ 2500 mAh). We call *network lifetime* to the elapsed time when the first node dies, without including the sink.

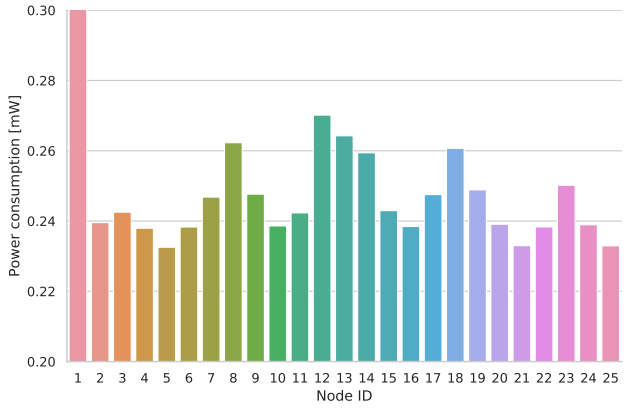
In Fig. 6.14 we show a bar plot with the mean values and 95% confidence intervals of such metric. We can see that WuR-based protocols present a higher lifetime than that of the traditional duty-cycled MAC approach in ContikiMAC. This means that the main goal of reducing the overall energy consumption with WuR is achieved. REFLOOD represents an improvement to 300% of ContikiMAC's lifetime in the best case, which is low traffic ($T = 1000$ s) and high WuR data rate (10 kbps). The low lifetime of ContikiMAC is explained by the high power consumption of the main radio. This is due to the numerous times spent in idle listening on the main radio and the need to transmit a preamble (which is a continuous transmission of the pending data packet) required for the protocol to work. On the other hand, the main radio of WuR-based protocols remains sleeping unless there is a packet generated at the application layer. Furthermore, thanks to the address included in the WuS, the nodes do not wake up the main radio in vain when other nodes are communicating. This removes the overhearing that is a typical problem in duty-cycled MAC protocols.

We can see easily that REFLOOD is the protocol that achieves the highest lifetime. There are some special cases, such as $T = 10$ s and $T = 60$ s for WuR data rate of 1 kbps, when PROPL presents a slightly higher lifetime than REFLOOD. However, its variability is higher than that of REFLOOD, because its value depends on the resulting DODAG, which can have different configurations for the same topology.

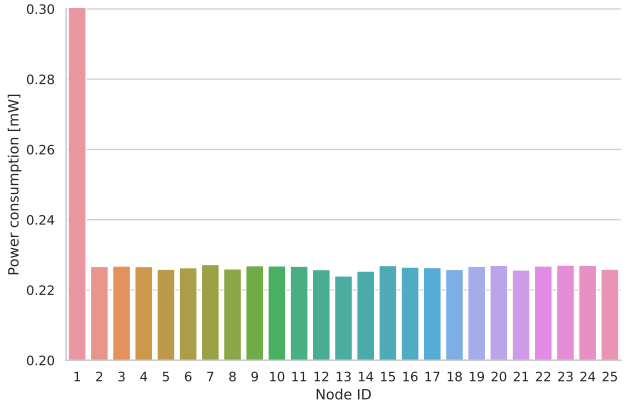
Besides, the WuR data rate has a significant impact on the result because it defines the amount of time over the air required to transmit a WuS, according to



(a) ContikiMAC

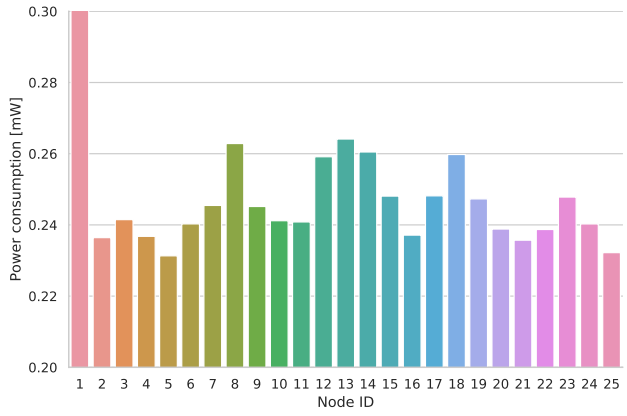


(b) PROPL

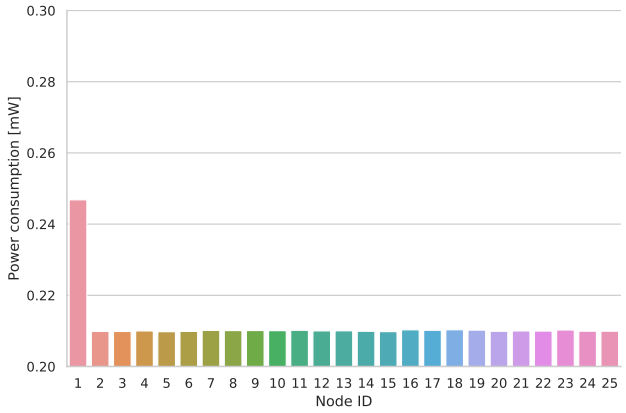


(c) REFLOOD

Figure 6.12: Load balancing at 1 kbps, $T = 1000$ s



(a) PROPL



(b) REFLOOD

Figure 6.13: Load balancing at 10 kbps, $T = 1000$ s

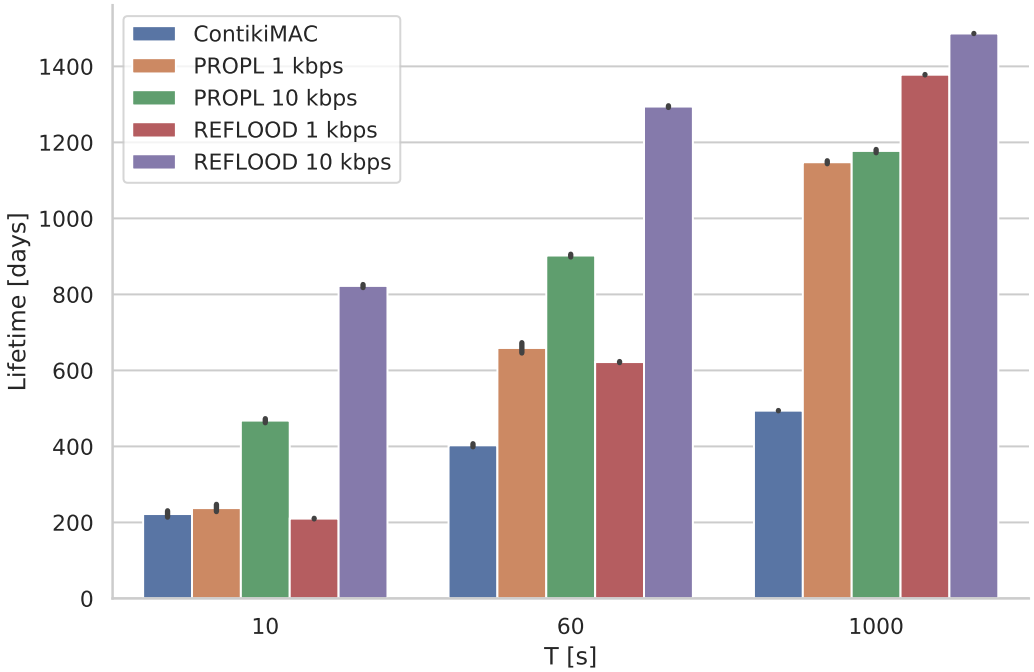


Figure 6.14: Network lifetime (higher is better)

Equation 6.3. Reducing the WuS size as much as possible is important because it compensates for the low data rate. Also, a high data rate can mitigate the low performance in high traffic scenarios. REFLOOD has a significant advantage because the size of the WuS is smaller than the other WuR-based protocol. Besides, the WuR consumption typically represents a significant portion of the overall power consumption of the device. When the WuS size is smaller, its duration in transmission mode is shorter, thus consuming less power. In the following section, we analyze the contributions of different states to the overall power consumption.

6.5.5 Power consumption breakdown

In this section, we analyze the power consumption breakdown of the most power-consuming node in different states (main node active CPU, main node sleep mode, main radio, and WuR). This helps to understand the states that contribute more to the lifetime behavior in each case. To illustrate it, we plot a stacked bar chart of a single repetition to see an example for each scenario in Fig. 6.15. *MCU* is the power consumption of the microcontroller (MCU) of the main node that performs the active tasks of the IoT device (managing sensors, computing algorithms, acquiring data, etc.). This is typically determined by the mean current consumption of the MCU in active mode. *Sleep* is the low power mode of the main node, i.e. when the aforementioned MCU is sleeping. *Main radio* is the power consumption of the main radio module and is the aggregation of its sleep, transmission, reception, and listening modes. Finally, *WuR* represents the power consumption of the WuR module and is the aggregation of its transmission, reception, and listening modes (when the WuR is in listening mode, the WuR microcontroller is sleeping). In

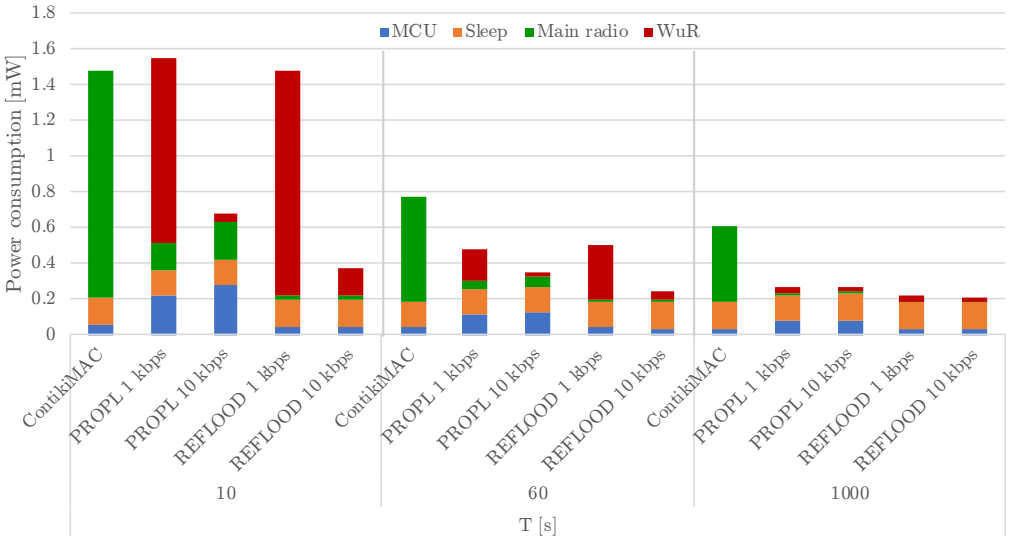


Figure 6.15: Power consumption breakdown of the most power-consuming node

this plot, we can see that the state that contributes the most to the overall power consumption in ContikiMAC is the main radio, for all traffic loads. Such observation is in line with the literature [7]. The main radio power consumption is significant because of the periodic idle listening which is independent of the traffic load. On the contrary, in PROPL, the contribution of the main radio is only significant in high traffic scenarios because of the rate of control packets sent by RPL which is higher in such cases, given that the elapsed time of the simulation is shorter. In contrast, such contribution in REFLOOD is not significant in any traffic load because it does not require control overhead on the main radio.

Notice that in low traffic scenarios, the relative importance of the WuR power consumption decreases. The reason for this is simply that the device spends a very short time in WuR TX mode (that is, transmitting a WuS) over an extremely long period of sleep mode. However, in the proactive case, the WuR contribution stays constant because of the control overhead that is required. PROPL has to send control packets regularly to maintain the routing structure, even using mechanisms to limit that traffic such as the Trickle algorithm [60], and each control message transmission is preceded by a WuS.

In conclusion, based on these results, it is better to use the WuR in low traffic, at the highest possible data rate, and with a reactive approach. In some particular cases, such as high-medium traffic scenarios ($T = 10$ s) and low data rates, PROPL may present a marginally higher lifetime but it comes at the cost of a complex implementation and control overhead, which requires more memory. PROPL's binary file occupies 47k bytes of flash memory out of the 48k of capacity measured in the MSP430, while REFLOOD occupies only 37k, as shown in Table 6.3. Furthermore, in extremely low traffic scenarios, the reactive approach can turn off the transceiver completely, as long as there is no packet generated at the application layer. Conversely, the proactive protocol requires some periodic maintenance with

Table 6.3: Memory size

Protocol	ROM (kilobytes)
PROPL	47.307
REFLOOD	37.257

control packets to keep track of the routing structure, even when there are no communications ongoing.

6.5.6 Random topology comparison

Figs. 6.16, 6.17, and 6.18 show the changes in PDR, latency, and network lifetime for the random topology of Fig. 6.8. The bars indicate the relative difference in percentage values between the results of the random topology and that of the grid topology of Fig. 6.7.

In all of them, we can see that the performance of REFLOOD and ContikiMAC is not degraded when running on the random topology. The main reason is that in ContikiMAC there is no routing because all the nodes are in the main radio range and communicate directly with the sink. In REFLOOD, the use of multiple paths mitigates the differences in WuS routes between sources and destination. On the other hand, PROPL is very sensitive to topology geometry. The location of the nodes impacts the resulting routing structure when it is not well balanced in the geometry. As a result, PROPL requires more retransmissions and reduces the PDR, increasing the latency, and reducing the lifetime.

In conclusion, REFLOOD is robust to topology changes because its performance remains the same when changing topologies, thanks to the use of multiple paths.

6.6 Conclusions

Wake-up radios is a revolutionary technology that promises to keep the simplicity of asynchronous MAC protocols with the performance close to precisely synchronized ones. In this chapter, we tackled one of the main challenges of this technology: the range mismatch between the secondary wake-up receiver and the main radio. Without this, the gain of this technology in terms of power consumption is not crystal clear considering realistic scenarios. In the previous work [21, 22], it was shown that relaying the WuS is mandatory to benefit from the maximum range of the main radio, and therefore achieve the best performance. In this chapter, we analyzed how to route the wake-up signal to wake up a destination node. The approaches can be classified between proactive and reactive. Based on our understanding of the WuR technology, the most suited applications for the WuR are those where the data traffic is low and event-driven [21, 41, 57, 58] because WuR requires a low data rate and is very sensitive to interferences and collisions on the communication channel. With that in mind, we designed a reactive routing protocol referred to as REFLOOD. To evaluate its performance we also implemented a proactive protocol based on RPL, called PROPL. We compared them by monitoring the packet delivery ratio, latency,

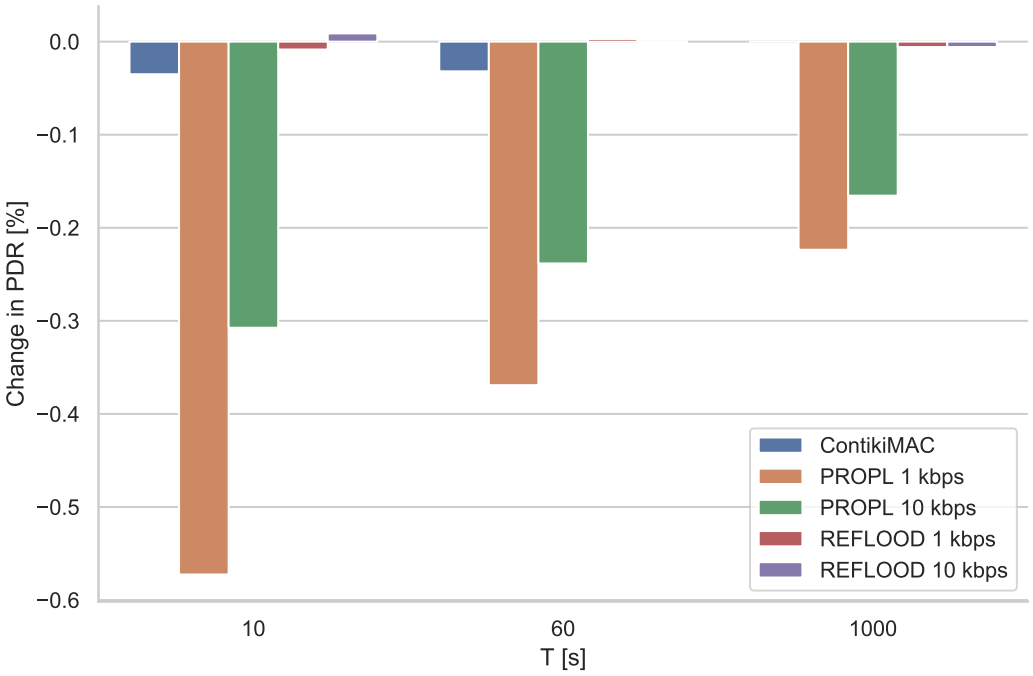


Figure 6.16: Differences in PDR between random and controlled topology

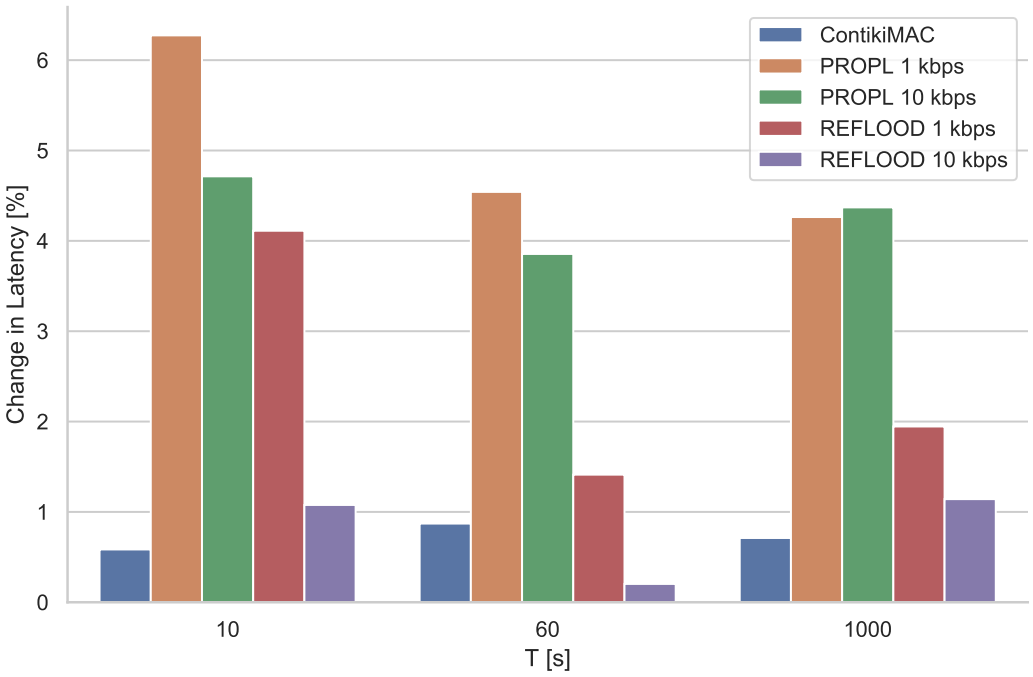


Figure 6.17: Differences in latency between random and controlled topology

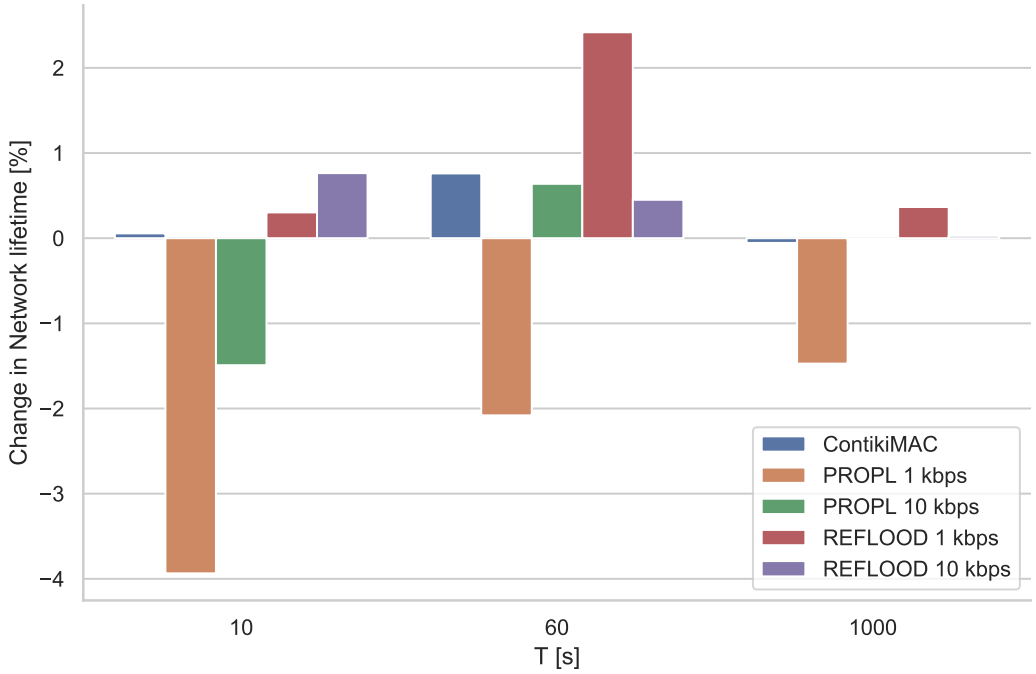


Figure 6.18: Differences in network lifetime between random and controlled topology

and power consumption, and the network lifetime based on simulations in COOJA with ContikiOS.

Our simulation results showed that REFLOOD improves the network lifetime up to 300% compared to traditional duty-cycled protocols. This WuR-based protocol has proven to be the best one based on our results. First, using a reactive approach allows wake-up signals to take multiple paths to reach the destination, which, in turn, improves reliability by eliminating the single point of failure and the funneling effect, compared to the proactive approach. Therefore, the protocol is more robust to different network topologies. As a consequence, the latency is also reduced because of the lower number of retransmissions required to deliver a data packet successfully. Besides, WuR is very sensitive to interferences and collisions, and REFLOOD can mitigate that problem by taking advantage of multiple paths. Second, this protocol allows reducing the size of the wake-up signal which contributes to reducing the power consumption. Hence, it minimizes the value of sync delay required to wake up a destination node, which in turn also reduces the latency. Finally, the lack of control overhead in reactive protocols is especially important to increase the network lifetime in low traffic applications with wake-up radios.

In the future, we plan to extend our work to support large-scale networks where the main source and main destination nodes are located multiple hops away on the main radio. In that case, the data frame will require multiple hops on the main radio. For this, we plan to segment the network into areas to limit the flooding and simply relay data frames between areas. Each one of those hops needs a routing strategy for the wake-up signal as analyzed in the current chapter. Moreover, we envision experiments at a large scale thanks to the FIT-IoT lab testbed [61].

Summary of the second contribution

In this part of the thesis, we focus on the problems at the network layer when using WuR. The first takeaway, in this case, is that this technology is particularly well suited for an opportunistic approach that addresses the issues of traditional routing infrastructures, such as RPL. Our solution, eLoBaPS takes advantage of that to extend the network lifetime thanks to load balancing. Moreover, the network is more stable and productive, which means that it takes less time to deliver the same number of packets than other solutions. The second takeaway is that our choice to focus on a reactive routing protocol has paid off, compared to proactive ones, for the challenge of wake-up signal routing, concerning the range mismatch. The use of multiple paths to reach a destination with the wake-up signal is the key to the success, and it is based on our conclusions from Part I, because we found that the use of acknowledgments on the wake-up signal was problematic. In particular, in low-traffic scenarios, WuR-based solutions clearly outperform traditional ones in terms of reliability, latency, and power consumption. Furthermore, we showed that the contribution of the communication protocol stack does not impact strongly the overall power consumption.

We designed an energy-aware cross-layer solution (eLoBaPS) to select opportunistic layer-3 parents. The nature of this protocol is more precise, providing more reproducibility than the implementation of RPL with W-MAC, and also more stable. It also extends the network lifetime up to 77% by consuming the battery of the feasible successors in a balanced way.

On the other hand, we showed how the reactive protocol that we designed, REFLOOD, provides multiple paths and almost no control overhead, improving the network lifetime in WSN applications up to 300% compared to traditional solutions. Moreover, this solution is simpler to deploy and maintain than proactive strategies.

Future works that continue the research on WuR should take these conclusions into account as a base to develop new protocols and ensure the success of this technology in IoT.

Conclusions

Chapter 7

Conclusion and future research directions

In this chapter, we present our final thoughts and conclusions about this research project, as well as some suggested research directions to continue to work towards the success of WuR in IoT.

7.1 Conclusion

The main goal of this thesis was to design a new protocol stack benefiting from wake-up radio technology to keep the advantages of an asynchronous protocol (such as low complexity, easy deployment, and operation) together with the performance of a synchronous protocol (high throughput and predictable reliability).

In the first part, we analyzed the MAC layer. We showed what is needed for this technology to work effectively and how it compares to traditional solutions in WSN. Furthermore, we analyzed the main problems that appear when using WuR: the range mismatch and the sync delay. We complemented this part with experiments on a state-of-the-art prototype to validate our simulations and gain more insights into the behavior of this technology, especially in noisy environments. In the second part, we proposed a cross-layer solution to address the issues at the network layer when using WuR. We successfully solved many traditional problems of the routing problem in WSN and improved the network lifetime to unprecedented levels thanks to an energy-aware solution that focuses on load balancing. We found that reactive strategies are more suited than proactive in WuR networks, and we designed and developed a protocol (REFLOOD) that addresses the main challenge of WuR routing: the range mismatch. At the same time, such protocol reduces the use of memory and is easy to implement and deploy because of its robustness to topology changes. The main limitation of WuR is that the performance is very sensitive to the packet error rate of the WuR medium. We have shown that using a coding scheme that forces transitions can significantly improve the robustness to interferences of the WuR links.

After all this work we can safely state that the WuR improves the network performance in WSN and can effectively extend the battery lifetime, both at the

MAC layer and the routing layer. We proved that it is possible to use asynchronous protocols with optimal performance in these types of applications. This is great news since these types of protocols are easier to implement and maintain, which is a big concern in the industry. We expected to have the best performance, in terms of reliability, in low traffic or delay-tolerant scenarios because the bit rate of WuR is very low. Fortunately, our results agreed with this statement and we found another reason why low traffic scenarios are interesting for WuR applications: the contribution of the communication stack is not significant to the overall power consumption. This way, in WuR-based solutions and low traffic scenarios, the radio module is not anymore the most significant source of power consumption.

7.2 Research directions

In this section, we discuss briefly some research directions to pursue new insights on WuR technologies and develop new ways to take advantage of them.

7.2.1 REFLOOD extension to support multi-hop networks

In this thesis, we proposed REFLOOD, a reactive routing protocol that allows WuS relaying in WuR networks. However, the design focused on a small network where all the nodes where one hop away from the sink on the main radio. In larger networks, the nodes may be further away requiring multiple hops of the data packet. Our design can be used to route the WuS and wake up an intermediate node that then forwards the data packet on the main radio. Nonetheless, this protocol has to be extended to select those intermediate nodes.

One option that could be explored is to organize the network in *areas*, as illustrated in Fig. 7.1. In the general case, there might be multiple sinks, also called *network gateways*. The areas are defined according to the distance of the nodes to these gateways, measured in number of hops. Nodes situated at the border of two different areas act as *border routers*, relaying data from one area to another. This way, the network is composed of network gateways, area border routers, and *legacy nodes*. The latter ones only transmit their data, generate, and relay WuS. A hybrid solution could be explored, where a proactive strategy is used to build the structure, and a reactive approach is used to route the WuS towards border routers to forward data packets.

In such a solution, the power consumption of the destination nodes would play an important role in the network lifetime. To minimize it, the determination of the sync delay, as presented in Chapter 6, is fundamental. An interesting approach would be to predict the sync delay with an artificial intelligence algorithm, such as reinforcement learning, based on previous communication attempts.

7.2.2 Mobility

When a node moves away from its original position, it needs to re-join the network and keep the routing structure in a valid state. The main challenge in mobility is to ensure the connectivity for all nodes in such a process. Currently, both LoBaPS and REFLOOD allow some degree of mobility in breadth, that is confined to each

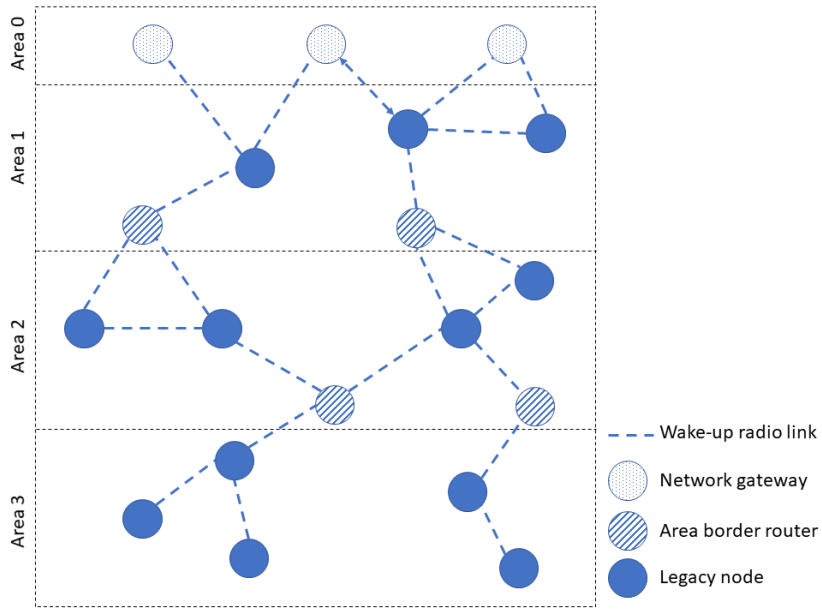


Figure 7.1: Network structure suggested

rank (in RPL-based LoBaPS) or area (in REFLOOD, as defined in Section 7.2.1), as a consequence of its design principle. However, if a node moves onto different depth levels, that is, crossing different ranks, a local repair is necessary to allow the mobile node to re-join the network and communicate.

Another option to investigate is to create a new type of packet associated with a new mechanism that allows to opportunistically forward data packets coming from mobile nodes, ignoring their rank. For this, we need to assume that a node can detect whenever it becomes a mobile node, for example, thanks to a movement sensor, such as an accelerometer. And then, we need to analyze how can we avoid routing loops or infinite flooding in that type of mechanism.

7.2.3 Large-scale experiments

We expect that the FIT-IoT testbed will soon integrate a new node that includes a WuR prototype. Thanks to that, it will be possible to automate large-scale experiments with multiple hops on the main radio. It is necessary for any further work that continues this analysis to perform experiments on real-world scenarios. Our experiments in Chapter 4 are a step towards this goal, by analyzing a single link with interferences and real-world noise. However, if it was possible to evaluate the designed protocols in a real network with many devices, we could be able to better understand other sources of problems related to the WuR range mismatch and interferences with other technologies.

The development of a hardware module that can be manufactured on large scale and modified after production can significantly improve such task. The design of the circuit must be carefully modeled to perform simulations and tests that are easy to reproduce. An option for such a design is the implementation of some of the WuR blocks in an FPGA. This way, new blocks can be tested to improve the

circuit without further changes to the physical module, accelerating the research path. Furthermore, it paves the way towards the adoption of artificial intelligence in higher layers of the communication stack.

Bibliography

[1] "Allied Market Research". In: (). URL: https://www.alliedmarketresearch.com/smart-cities-market.

[2] Fayez Alfayez, Mohammad Hammoudeh, and Abdelrahman Abuarqoub. "A Survey on MAC Protocols for Duty-cycled Wireless Sensor Networks". In: *Procedia Computer Science* 73 (2015). International Conference on Advanced Wireless Information and Communication Technologies (AWICT 2015), pp. 482 –489. ISSN: 1877-0509.

[3] R. Piyare, A. L. Murphy, C. Kiraly, P. Tosato, and D. Brunelli. "Ultra Low Power Wake-Up Radios: A Hardware and Networking Survey". In: *IEEE Communications Surveys Tutorials* 19.4 (2017), pp. 2117–2157.

[4] Klaus Schwab. *The Fourth Industrial Revolution*. en. Google-Books-ID: ST_FDAAAQBAJ. Crown, Jan. 2017. ISBN: 978-1-5247-5887-5.

[5] Wesley Doorsamy. *The Disruptive Fourth Industrial Revolution*. en. Springer Nature. ISBN: 978-3-030-48230-5.

[6] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. "Internet of Things for Smart Cities". In: *IEEE Internet of Things Journal* 1.1 (2014), pp. 22–32. ISSN: 2327-4662.

[7] C. Cano, B. Bellalta, A. Sfairopoulou, and M. Oliver. "Low energy operation in WSNs: A survey of preamble sampling MAC protocols". en. In: *Computer Networks* 55.15 (Oct. 2011), pp. 3351–3363. ISSN: 1389-1286. DOI: 10.1016/j.comnet.2011.06.022. URL: http://www.sciencedirect.com/science/article/pii/S1389128611002349 (visited on 06/24/2020).

[8] T. Watteyne, J. Weiss, L. Doherty, and J. Simon. "Industrial IEEE802.15.4e networks: Performance and trade-offs". In: *2015 IEEE International Conference on Communications (ICC)*. June 2015, pp. 604–609. DOI: 10.1109/ICC.2015.7248388.

[9] A. Pegatoquet, T. N. Le, and M. Magno. "A Wake-Up Radio-Based MAC Protocol for Autonomous Wireless Sensor Networks". In: *IEEE/ACM Transactions on Networking* 27.1 (Feb. 2019), pp. 56–70. ISSN: 1063-6692. DOI: 10.1109/TNET.2018.2880797.

[10] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. "Cross-Level Sensor Network Simulation with COOJA". In: *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*. ISSN: 0742-1303. Nov. 2006, pp. 641–648. DOI: 10.1109/LCN.2006.322172.

[11] A. Dunkels, B. Gronvall, and T. Voigt. "Contiki - a lightweight and flexible operating system for tiny networked sensors". In: *29th Annual IEEE International Conference on Local Computer Networks*. ISSN: 0742-1303. Nov. 2004, pp. 455–462. DOI: 10.1109/LCN.2004.38.

[12] T. Winter. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*. RFC 6550. RFC Editor, 2012.

[13] Nathan Michael Pletcher, Ultra-low Power, Wake-up Receivers, Nathan Michael Pletcher, and Nathan Michael Pletcher. *Ultra-Low Power Wake-Up Receivers for Wireless Sensor Networks*. 2008.

[14] C. Salazar, A. Cathelin, A. Kaiser, and J. Rabaey. "A 2.4 GHz Interferer-Resilient Wake-Up Receiver Using A Dual-IF Multi-Stage N-Path Architecture". In: *IEEE Journal of Solid-State Circuits* 51.9 (Sept. 2016), pp. 2091–2105. ISSN: 0018-9200. DOI: 10.1109/JSSC.2016.2582509.

[15] Anjana Dissanayake, Henry L. Bishop, Jesse Moody, Henry Muhlbauer, Benton H. Calhoun, and Steven M. Bowers. "A Multichannel, MEMS-Less -99dBm 260nW Bit-Level Duty Cycled Wakeup Receiver". In: *2020 IEEE Symposium on VLSI Circuits*. ISSN: 2158-5636. June 2020, pp. 1–2. DOI: 10.1109/VLSICircuits18222.2020.9162785.

[16] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. Popovici, and L. Benini. "Design, Implementation, and Performance Evaluation of a Flexible Low-Latency Nanowatt Wake-Up Radio Receiver". In: *IEEE Trans. Ind. Informat.* 12.2 (2016), pp. 633–644. ISSN: 1551-3203.

[17] Rajeev Piyare, Timofei Istomin, and Amy L. Murphy. "WaCo: A Wake-Up Radio COOJA Extension for Simulating Ultra Low Power Radios". In: *EWSN*. 2017.

[18] Dhongue Lee and Patrick P. Mercier. "Introduction to Ultra Low Power Transceiver Design". en. In: *Ultra-Low-Power Short-Range Radios*. Ed. by Patrick P. Mercier and Anantha P. Chandrakasan. Integrated Circuits and Systems. Cham: Springer International Publishing, 2015, pp. 1–23. ISBN: 978-3-319-14714-7. DOI: 10.1007/978-3-319-14714-7_1. URL: https://doi.org/10.1007/978-3-319-14714-7_1 (visited on 07/10/2020).

[19] Stefano Basagni, Federico Ceccarelli, Chiara Petrioli, Nithila Raman, Abhimanyu V Sheshashayee, and ECE Dept. "Wake-up Radio Ranges: A Performance Study". en. In: (), p. 6.

[20] R. Su, T. Watteyne, and K. S. J. Pister. "Comparison between Preamble Sampling and Wake-Up Receivers in Wireless Sensor Networks". In: *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*. 2010, pp. 1–5.

[21] S. L. Sampayo, J. Montavont, F. Prégaldiny, and T. Noël. "Is Wake-Up Radio the Ultimate Solution to the Latency-Energy Tradeoff in Multi-hop Wireless Sensor Networks?" In: *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2018, pp. 1–8. DOI: 10.1109/WiMOB.2018.8589163.

[22] S. Basagni, C. Petrioli, and D. Spenza. "CTP-WUR: The collection tree protocol in wake-up radio WSNs for critical applications". In: *2016 International Conference on Computing, Networking and Communications (ICNC)*. 2016, pp. 1–6. DOI: 10.1109/ICCNC.2016.7440687.

[23] Falko Dressler et al. "Monitoring Bats in the Wild: On Using Erasure Codes for Energy-Efficient Wireless Sensor Networks". In: *ACM Trans. Sen. Netw.* 12.1 (Feb. 2016), 7:1–7:29. ISSN: 1550-4859. DOI: 10.1145/2875426.

[24] N. Simon et al. "Indoor localization system for emergency responders with ultra low-power radio landmarks". In: *2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*. 2015, pp. 309–314. DOI: 10.1109/I2MTC.2015.7151285.

[25] Andras Varga. "OMNeT++". In: *Modeling and Tools for Network Simulation*. Ed. by Klaus Wehrle, Mesut Güneş, and James Gross. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–59. ISBN: 978-3-642-12331-3.

[26] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl. "Has Time Come to Switch From Duty-Cycled MAC Protocols to Wake-Up Radio for Wireless Sensor Networks?" In: *IEEE/ACM Transactions on Networking* 24.2 (2016), pp. 674–687. ISSN: 1063-6692.

[27] Timo Kumberg, Marc Schink, Leonhard Reindl, and Christian Schindelhauer. "T-ROME: A Simple and Energy Efficient Tree Routing Protocol for Low-Power Wake-up Receivers". In: (Feb. 2017).

[28] Georgia Koutsandria, Valerio Di Valerio, Dora Spenza, Stefano Basagni, and Chiara Petrioli. "Wake-up radio-based data forwarding for green wireless networks". en. In: *Computer Communications* 160 (July 2020), pp. 172–185. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2020.05.046. URL: http://www.sciencedirect.com/science/article/pii/S0140366420300128 (visited on 07/28/2020).

[29] D. Ghose and F. Y. Li. "Enabling Backoff for SCM Wake-Up Radio: Protocol and Modeling". In: *IEEE Communications Letters* 21.5 (May 2017), pp. 1031–1034. ISSN: 1089-7798. DOI: 10.1109/LCOMM.2017.2653779.

[30] D. Ghose, F. Y. Li, and V. Pla. "MAC Protocols for Wake-Up Radio: Principles, Modeling and Performance Analysis". In: *IEEE Transactions on Industrial Informatics* 14.5 (May 2018), pp. 2294–2306. ISSN: 1551-3203. DOI: 10.1109/TII.2018.2805321.

[31] Abhimanyu Venkatraman Sheshashayee and Stefano Basagni. "Multi-Hop Wake-Up Radio Relaying for the Collection Tree Protocol". In: *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*. ISSN: 2577-2465, 1090-3038. Sept. 2019, pp. 1–6. DOI: 10.1109/VTCFall.2019.8891447.

[32] Stefano Basagni, Chiara Petrioli, and Dora Spenza. "CTP-WUR: The collection tree protocol in wake-up radio WSNs for critical applications". In: *2016 International Conference on Computing, Networking and Communications (ICNC)*. Feb. 2016, pp. 1–6. DOI: 10.1109/ICCNC.2016.7440687.

[33] Fayçal Ait Aoudia, Matthieu Gautier, and Olivier Berder. "OPWUM: Opportunistic MAC Protocol Leveraging Wake-Up Receivers in WSNs". In: *Journal of Sensors* (Jan. 2016). DOI: 10.1155/2016/6263719. URL: https://hal.inria.fr/hal-01244800.

[34] "The Network Simulator - ns-3". In: (2009). URL: https://www.nsnam.org/overview/what-is-ns-3/.

[35] David Benedetti, Chiara Petrioli, and Dora Spenza. "GreenCastalia: An Energy-Harvesting-Enabled Framework for the Castalia Simulator". In: *Proceedings of ACM ENSSys 2013*. Rome, Italy: ACM, 2013, 7:1–7:6.

[36] "T-Mote Sky Datasheet". In: (). URL: http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf.

[37] Adam Dunkels, Fredrik Österlind, and Zhitao He. "An Adaptive Communication Architecture for Wireless Sensor Networks". In: *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*. SenSys '07. Sydney, Australia: ACM, 2007, pp. 335–349. ISBN: 978-1-59593-763-6. DOI: 10.1145/1322263.1322295.

[38] Adam Dunkels. "The ContikiMAC radio duty cycling protocol". In: (May 2012).

[39] Adam Dunkels, Joakim Eriksson, Niclas Finne, and Nicolas Tsiftes. "Powertrace: Network-level Power Profiling for Low-power Wireless Networks". In: (Apr. 2011).

[40] Stefano Basagni, Federico Ceccarelli, Chiara Petrioli, Nithila Raman, Abhimanyu V Sheshashayee, and ECE Dept. "Wake-up Radio Ranges: A Performance Study". In: (), p. 6.

[41] Sebastian L Sampayo, Julien Montavont, and Thomas Noel. "LoBaPS: Load Balancing Parent Selection for RPL Using Wake-Up Radios". In: *2019 IEEE Symposium on Computers and Communications (ISCC) (IEEE ISCC 2019)*. Barcelona, Spain, June 2019.

[42] Shahin Farahani. *ZigBee Wireless Networks and Transceivers*. en. Elsevier, 2008. ISBN: 978-0-7506-8393-7. DOI: 10.1016/B978-0-7506-8393-7.X0001-5. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780750683937X00015 (visited on 11/11/2019).

[43] "Linx Technologies, ANT-868-CW-RCS". In: (). URL: https://www.mouser.fr/datasheet/2/238/ant-868-cw-rcs-1659515.pdf.

[44] T. Polonelli, M. Magno, and L. Benini. "Poster Abstract: An Ultra-Low Power Wake up Radio with Addressing and Retransmission Capabilities for Advanced Energy Efficient MAC Protocols". In: *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. Apr. 2016, pp. 1–2. DOI: 10.1109/IPSN.2016.7460696.

[45] Debasish Ghose, Anders Frøytlog, and Frank Y. Li. "Enabling Early Sleeping and Early Data Transmission in Wake-up Radio-enabled IoT Networks". In: *Computer Networks* 153 (Mar. 2019). DOI: 10.1016/j.comnet.2019.03.002.

[46] P. Thubert. *Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)*. RFC 6552. RFC Editor, 2012.

[47] O. Iova, F. Theoleyre, and T. Noel. "Stability and efficiency of RPL under realistic conditions in Wireless Sensor Networks". In: *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. Sept. 2013, pp. 2098–2102.

[48] S. N. Shukla and T. A. Champaneria. "Survey of various data collection ways for smart transportation domain of smart city". In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. Feb. 2017, pp. 681–685.

[49] Arthur Benjamin, Gary Chartrand, and Ping Zhang. *The Fascinating World of Graph Theory*. en. June 2017. ISBN: 978-0-691-17563-8. URL: https://press.princeton.edu/books/paperback/9780691175638/the-fascinating-world-of-graph-theory (visited on 01/06/2021).

[50] A. Dunkels, B. Gronvall, and T. Voigt. "Contiki - a lightweight and flexible operating system for tiny networked sensors". In: *29th Annual IEEE International Conference on Local Computer Networks*. 2004, pp. 455–462.

[51] Thomas Clausen, Jiazi Yi, and Ulrich Herberg. "Lightweight On-demand Ad hoc Distance-vector Routing - Next Generation (LOADng): Protocol, extension, and applicability". en. In: *Computer Networks* 126 (Oct. 2017), pp. 125–140. ISSN: 13891286. DOI: 10.1016/j.comnet.2017.06.025. URL: https://linkinghub.elsevier.com/retrieve/pii/S1389128617302694 (visited on 01/15/2020).

[52] Jiazi Yi, Thomas Clausen, and Yuichi Igarashi. "Evaluation of routing protocol for low power and Lossy Networks: LOADng and RPL". In: *2013 IEEE Conference on Wireless Sensor (ICWISE)*. Kuching, Sarawak, Malaysia: IEEE, Dec. 2013, pp. 19–24. ISBN: 978-1-4799-1576-7 978-1-4799-1560-6. DOI: 10.1109/ICWISE.2013.6728773. URL: http://ieeexplore.ieee.org/document/6728773/ (visited on 01/21/2020).

[53] Jiazi Yi and Thomas Clausen. "Collection Tree Extension of Reactive Routing Protocol for Low-Power and Lossy Networks:" en. In: *International Journal of Distributed Sensor Networks* (Mar. 2014). DOI: 10.1155/2014/352421. URL: https://journals.sagepub.com/doi/10.1155/2014/352421 (visited on 01/15/2020).

[54] Joydeep Tripathi, Jaudelice C. de Oliveira, and J. P. Vasseur. "Proactive versus reactive routing in low power and lossy networks: Performance analysis and scalability improvements". en. In: *Ad Hoc Networks* 23 (Dec. 2014), pp. 121–144. ISSN: 1570-8705. DOI: 10.1016/j.adhoc.2014.06.007. URL: http://www.sciencedirect.com/science/article/pii/S1570870514001243 (visited on 05/27/2020).

[55] Axel Neumann, Ester López, and Leandro Navarro. "Evaluation of mesh routing protocols for wireless community networks". en. In: *Computer Networks*. Community Networks 93 (Dec. 2015), pp. 308–323. ISSN: 1389-1286. DOI: 10.1016/j.comnet.2015.07.018. URL: http://www.sciencedirect.com/science/article/pii/S1389128615002522 (visited on 05/27/2020).

[56] José V. V. Sobral, Joel J. P. C. Rodrigues, Ricardo A. L. Rabêlo, Jalal Al-Muhtadi, and Valery Korotaev. "Routing Protocols for Low Power and Lossy Networks in Internet of Things Applications". en. In: *Sensors* 19.9 (Jan. 2019), p. 2144. DOI: 10.3390/s19092144. URL: https://www.mdpi.com/1424-8220/19/9/2144 (visited on 01/15/2020).

[57] Sebastian L. Sampayo, Julien Montavont, and Thomas Noël. "eLoBaPS: Towards Energy Load Balancing with Wake-Up Radios for IoT". en. In: *Ad-Hoc, Mobile, and Wireless Networks*. Ed. by Maria Rita Palattella, Stefano Scanzio, and Sinem Coleri Ergen. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 388–403. ISBN: 978-3-030-31831-4. DOI: 10.1007/978-3-030-31831-4_27.

[58] Sebastian Lucas Sampayo, Julien Montavont, and Thomas Noel. "A Performance Study of the Behavior of the Wake-Up Radio in Real-World Noisy Environments". In: *AWAKE workshop of the 2020 International Conference on Embedded Wireless Systems and Networks on Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks*. AWAKE workshop of EWSN '20. Lyon, France: Junction Publishing, Feb. 2020, pp. 206–211. ISBN: 978-0-9949886-4-5. (Visited on 06/23/2020).

[59] "IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer". In: *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)* (Apr. 2012), pp. 1–225. DOI: 10.1109/IEEESTD.2012.6185525.

[60] Philip Levis and Thomas Heide Clausen. *The Trickle Algorithm*. en. Library Catalog: tools.ietf.org. URL: https://tools.ietf.org/html/rfc6206 (visited on 07/09/2020).

[61] C. et al. Adjih. "FIT IoT-LAB: A large scale open experimental IoT testbed". In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. Dec. 2015, pp. 459–464.

List of Figures

List of Tables

Suite de protocoles de réseau polymorphe dans les réseaux hétérogènes Wake-up Radio IoT

Résumé

Chaque année, de plus en plus d'appareils se connectent à l'internet dans différents domaines de la vie tels que les bâtiments intelligents et les transports intelligents. Le marché mondial des villes intelligentes était évalué à cinq cents milliards de dollars en 2017 et devrait atteindre deux mille milliards d'ici 2025 [1]. Les réseaux de capteurs sans fil (WSN) sont couramment utilisés pour de telles applications lorsqu'il est nécessaire de mesurer une variable physique de l'environnement et de rendre les informations disponibles sur l'internet. Les nœuds destinés à cet objectif comprennent des dispositifs à faible puissance et à ressources limitées ayant une portée de communication à distance limitée. Pour couvrir des zones plus étendues, ces nœuds s'interconnectent sans fil dans ce que l'on appelle le WSN multi-hop. Le principal défi dans ces applications est de prolonger autant que possible la durée de vie de la batterie, ce qui se traduit par une réduction de la consommation d'énergie tout en conservant des performances de réseau de bonne qualité.

Traditionnellement, la consommation d'énergie était contrôlée dans ces réseaux par une forme de cycle de service dans le protocole de communication au niveau de la couche MAC, qui échangeait la latence contre l'efficacité énergétique. Ces dernières années, la technologie Wake-Up Radio (WuR) a progressé avec une acceptation croissante car elle promet la fin de ce compromis [3]. Le WuR est un récepteur secondaire qui écoute le canal en permanence pendant que l'émetteur-récepteur principal reste en sommeil et ne se réveille qu'à la demande par un signal sur le canal WuR. Ensuite, les données sont transmises par la radio principale. Les éléments essentiels sont expliqués au

chapitre 2.

Le but de cette thèse est de concevoir une nouvelle pile de protocoles bénéficiant de la technologie radio de réveil pour conserver les avantages d'un protocole asynchrone (tels que la faible complexité, la facilité de déploiement et d'exploitation) ainsi que les performances d'un protocole synchrone (haut débit et fiabilité prévisible). La partie I de cette thèse présente notre recherche sur les radios de réveil à la couche MAC ainsi que notre première contribution principale sur un nouveau protocole MAC basé sur les radios de réveil. Nous présentons également les simulations et les expériences sur un prototype réel que nous avons réalisé pour l'évaluer. Ensuite, nous analysons dans la partie II l'impact de cette technologie sur la couche de routage. Le principal problème est que la portée de la WuR est plus courte que celle de la radio principale. Par conséquent, l'utilisation du WuR conduit à des réseaux très denses qui peuvent augmenter le nombre de sauts nécessaires pour communiquer un paquet de données. C'est dans cet esprit que nous avons conçu REFLOOD, un protocole de routage réactif, pour résoudre le problème de décalage de portée.

Introduction

Les recherches menées dans le cadre de cette thèse ont eu lieu entre février 2018 et janvier 2021. Notre principal objectif était la conception de protocoles de communication pour une nouvelle technologie sans fil de l'Internet des objets (IoT) : Wake-up Radios (WuR). Comme l'indique le titre, il s'agit d'un *suite* de protocoles car il comprend de nombreuses couches de la pile réseau. Il est *polymorphe* car ils ont un comportement hybride qui peut changer en fonction de l'état actuel du réseau. Enfin, il est *hétérogène* parce qu'ils sont conçus pour les dispositifs IdO qui utilisent simultanément plusieurs technologies radio sans fil. Ce travail fait partie du projet WakeUp, fondé par l'Agence Nationale de la Recherche (ANR), où nous avons collaboré avec un partenaire académique (Université de Rennes 1), un institut de recherche (CEA-LETI), et une startup (Wi6Labs).

Dans le domaine des réseaux de capteurs sans fil (WSN), l'objectif principal est de réduire au maximum la consommation d'énergie. Traditionnellement, la solution consiste à utiliser des mécanismes de cycle d'obligation au niveau de la couche de contrôle d'accès au milieu (MAC). Cependant, il y a un compromis à faire car si la consommation d'énergie est fortement réduite, la latence devient alors élevée. De nombreuses applications dans le domaine industriel nécessitent à la fois une efficacité énergétique et une faible latence, par exemple dans les boucles de contrôle avec des capteurs et des actionneurs. La technologie des wake-up radios promet la fin de ce compromis en réduisant

les deux amplitudes en même temps.

La plupart des travaux réalisés jusqu'à présent en vue du développement des wake-up radios ont porté sur l'aspect matériel, et l'on n'a pas accordé trop d'attention à l'aspect réseau.

Contexte

Dans les sections suivantes, nous fournissons un contexte général pour comprendre l'origine de cette thèse.

The Internet of Things

Ces dernières années, de nombreux auteurs se sont accordés sur le fait que le monde est en transition vers une nouvelle révolution industrielle [4], [5]. Non seulement à cause des moteurs numériques tels que l'intelligence artificielle (IA) et l'Internet des objets (IoT), mais aussi à cause des biotechnologies qui permettent la biologie synthétique, et le séquençage des gènes, pour n'en citer que quelques-uns. La quatrième révolution industrielle transforme la vie des gens grâce à de nouvelles méthodes de travail et d'interaction, comme jamais auparavant l'humanité n'en a connu. La vitesse de ces changements est désormais exponentielle au lieu d'être linéaire par rapport à la révolution précédente. L'évolution est si profonde qu'elle modifie notre identité par un changement de paradigme, passant d'une société de l'information et de la communication à une société superintelligente [5].

La révolution agraire, il y a 10 000 ans, avec la domestication des animaux, a permis aux gens de passer de la recherche de nourriture à l'agriculture. Cela a profondément changé la société qui est passée d'un style nomade à des établissements plus importants, et le début de l'urbanisation et des villes. La première révolution industrielle, entre le XVIIIe et le XIXe siècle, sous l'impulsion des chemins de fer et de la machine à vapeur, a entraîné un passage de la production manuelle à la production mécanique. Quelques décennies plus tard, à la fin du XIXe siècle, la deuxième révolution industrielle a permis une production de masse grâce à l'électricité et aux chaînes de montage. Dans la décennie 1960, la troisième révolution industrielle a commencé avec l'invention des transistors MOS, qui a donné naissance aux ordinateurs et à l'Internet.

Aujourd'hui, depuis le début du 21e siècle, la quatrième révolution industrielle se caractérise par la large diffusion de l'Internet, qui ne se limite pas aux ordinateurs terminaux, mais qui relie tous les objets du monde physique au monde numérique. Des objets qui ne faisaient partie que de l'environnement physique au siècle dernier et qui ne pouvaient interagir avec l'homme que manuellement sont maintenant équipés d'un matériel informatique exécutant un programme logiciel, avec une connectivité au monde cybernétique. Ces

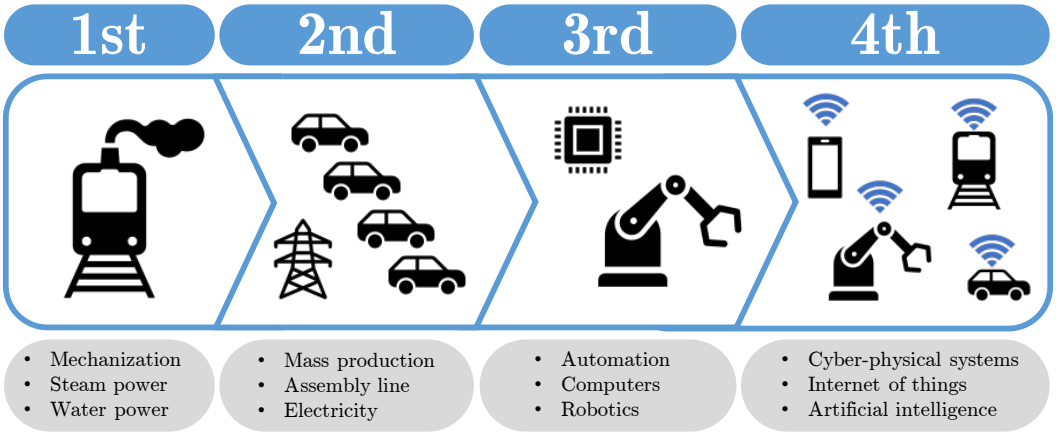


Figure 2: Révolutions industrielles au cours de l'histoire

objets peuvent maintenant interagir entre eux, dans ce que l'on appelle la communication "d'appareil à appareil" ou "de machine à machine", pour automatiser davantage les processus de notre vie quotidienne. Au début, cette connectivité a été câblée avec la même infrastructure que l'internet initial. Cependant, avec les applications croissantes des systèmes cyberphysiques, les systèmes de communication sans fil offrent des moyens plus souples de fournir une connectivité omniprésente et mobile aux choses de la vie quotidienne. C'est la définition de l'internet des objets (IoT). L'évolution de cette voie de l'industrialisation de l'humanité est décrite dans Fig. 2.

Un sous-domaine particulier de l'IdO est celui des réseaux de capteurs sans fil (WSN). Les nœuds de ces réseaux comprennent des systèmes embarqués à faible puissance et à ressources limitées, dotés de capacités de communication sans fil.

Réseaux de capteurs sans fil

L'objectif du WSN dans l'IdO est de mesurer les variables physiques de l'environnement et de les rendre disponibles sur Internet. En outre, des commandes peuvent être envoyées depuis l'internet vers des actionneurs pour modifier l'environnement physique.

Dans WSN, les nœuds sont de petits dispositifs dont la puissance et les capacités de calcul sont limitées. Chaque nœud est équipé d'un ou plusieurs capteurs qui mesurent une variable physique de l'environnement. À l'aide d'un simple microcontrôleur, le dispositif peut traiter ces informations et, à l'aide d'une radio sans fil, il transmet les données à un nœud spécial, appelé *sink* ou *gateway*. Le puits est chargé de collecter les données de tous les capteurs du réseau, de les traiter et éventuellement de les connecter à l'Internet. Le

schéma de trafic généré à cette fin est communément appelé *convergecast* ou *multi-point-to-point*. C'est le principal modèle que nous aborderons dans cette thèse.

Le fait que les ressources de ces dispositifs soient limitées se traduit par une portée de communication restreinte. Ensuite, pour couvrir une zone plus large, les nœuds s'interconnectent entre eux dans ce que l'on appelle le réseau *multi-saut*. Les nœuds intermédiaires, dans ce cas, relaient les paquets de données de ceux qui sont plus éloignés.

Il existe de nombreux exemples autour de cette idée, tels que la ville intelligente, l'usine intelligente ou le bâtiment intelligent [6]. Dans un bâtiment, il y a beaucoup de tâches qui sont traditionnellement effectuées manuellement, avec un travail humain. Depuis la plus simple, comme allumer et éteindre les lumières avec un interrupteur, jusqu'à des tâches plus complexes, comme détecter une fuite d'eau ou un dommage structurel. La collecte de données par le biais du WSN nous permet d'être proactifs, c'est-à-dire de prendre des mesures avant qu'un tel événement ne se produise. Cela réduit les coûts de maintenance et permet une utilisation plus efficace des ressources. En outre, nous pouvons également combiner les mesures avec des algorithmes d'intelligence artificielle pour prévoir ce qui va se passer à l'avenir et prendre de meilleures décisions en conséquence.

L'un des principaux défis du WSN dans les années à venir est de prolonger la durée de vie de la batterie du réseau, tout en maintenant une bonne qualité de communication, en répondant aux exigences de l'application. La raison en est que le remplacement manuel fréquent des batteries de milliers d'appareils n'est pas possible [7], [8], [9].

Motivations

Chaque année, dans le monde entier, plus de 15 milliards de piles sont jetées, contenant des matériaux toxiques qui contaminent l'environnement et créent de graves problèmes de santé pour la population. En conséquence, cela augmente les coûts médicaux associés à la guérison de ces problèmes. En outre, les grands déploiements peuvent consister en jusqu'à dix mille dispositifs de détection, où les coûts logistiques de remplacement des batteries sont énormes. Cela constitue un énorme obstacle à l'adoption de nouvelles technologies qui peuvent améliorer considérablement l'efficacité dans de nombreuses industries. C'est pourquoi nous devons réduire et, à terme, éliminer l'utilisation des piles.

Traditionnellement, le sous-système le plus énergivore d'un nœud de capteur est le module émetteur-récepteur [7]. Par conséquent, la consommation d'énergie est réduite dans ces dispositifs en recyclant l'activité au niveau de la couche de contrôle d'accès au support (MAC) de la pile du réseau [7]. Ainsi, le

nœud ne peut communiquer que pendant une courte période active, alors que la plupart du temps l'émetteur-récepteur est en mode veille. La communication avec ces dispositifs peut être réalisée *synchroniquement*, en coordonnant les créneaux horaires d'émission et de réception pour chaque nœud, ou *asynchronement*, lorsque les nœuds ne s'accordent sur aucun horaire particulier. Les protocoles MAC synchrones se sont avérés très efficaces dans les scénarios de trafic élevé et les réseaux stables [8]. Toutefois, dans de nombreuses applications WSN, la charge de trafic est faible et le réseau est dynamique. Dans ces cas, les protocoles synchrones présentent beaucoup de surcharge et leurs mises en œuvre, déploiements et opérations restent difficiles. Les protocoles MAC asynchrones sont mieux adaptés à ces applications car ils ne suivent aucun calendrier et ne font aucune hypothèse sur le réseau. En conséquence, le dispositif doit vérifier périodiquement le canal en introduisant une écoute au repos (lorsqu'il n'y a pas d'autre nœud qui doit communiquer) ou surréception (lorsque l'appareil n'est pas la destination prévue de la communication) des phénomènes qui peuvent avoir un impact significatif sur la consommation de la batterie. En outre, les collisions et le surcontrôle des protocoles gaspillent de l'énergie dans les transmissions que nous voulons réduire autant que possible. Enfin, la durée de la période active entraîne un compromis entre latence et efficacité énergétique dans ce type de solutions, en raison du temps limité pour communiquer à chaque cycle.

Récemment, la technologie Wake-up Radios (WuR) a progressé en tant que nouvelle solution pour prolonger la durée de vie du réseau WSN [3]. Sa consommation d'énergie ultra-faible et sa fonction de mise en marche permanente permettent de surmonter les problèmes des protocoles MAC asynchrones traditionnels à cycle obligatoire. Le WuR est un récepteur secondaire qui écoute le canal en permanence pendant que l'émetteur-récepteur principal reste en veille. Lorsqu'un nœud veut communiquer un paquet de données, il envoie d'abord un signal de réveil (un paquet sur le canal WuR) vers la destination. À la réception, la destination réveille la radio principale et attend en mode d'écoute le paquet de données. Après avoir échangé des données et un accusé de réception, la destination remet sa radio principale en mode veille pour économiser l'énergie. Une introduction détaillée au WuR est présentée dans la section 2. Cette technologie est révolutionnaire car elle promet la fin de l'écoute en repos et de la surréception à la couche MAC des protocoles asynchrones, et pas d'algorithmes synchronisés plus précis.

Une des principales limites de la WuR est une faible sensibilité, qui se traduit par un décalage de portée entre la radio principale (longue ou moyenne portée, de l'ordre de 200 m) et la WuR (courte portée, environ 20 m) [9]. Par conséquent, l'utilisation du WuR conduit à des réseaux très denses, car les

nœuds doivent être proches les uns des autres, pour communiquer par le biais de la très courte portée du WuR. Par conséquent, le réseau devient une architecture à deux niveaux dans laquelle nous devons traiter avec deux ensembles de voisins (un pour chaque interface radio) qui se superposent potentiellement. Ce problème est crucial pour la diffusion de la technologie wake-up radio, et il n'a pas encore été étudié de manière approfondie.

C'est pourquoi nous mettons en évidence le défi de recherche suivant dans cette thèse :

Défi Scientifique

Peut-on atteindre avec les protocoles asynchrones basés sur WuR le même niveau de performance que celui des protocoles synchrones, avec tous les avantages des premiers ? Comment gérer une telle architecture à deux niveaux, à la fois au niveau du MAC et du réseau ?

Goal and main contributions

L'objectif principal de cette thèse est d'explorer les avantages de la technologie wake-up radio au niveau de la couche MAC et de la couche de routage, dans les applications WSN. Le principal avantage du WuR se situe au niveau de la couche MAC, car il élimine le cycle d'obligation et réduit en même temps la latence et la consommation d'énergie. Il permet d'utiliser des protocoles purement asynchrones, qui sont plus faciles à mettre en œuvre, à déployer et à exploiter que les protocoles synchrones traditionnels, et cela avec le même niveau de performance. En ce qui concerne le modèle OSI, la couche de routage doit être agnostique par rapport à la technologie sous-jacente utilisée dans les couches inférieures. Cependant, le WuR introduit un réseau secondaire, en parallèle avec le réseau primaire, avec ses propres liens. Le calcul des itinéraires sur le multi-graphe créé par les liaisons de la radio principale et du WuR est une tâche difficile et n'a pas encore été étudié. Nous nous intéressons en particulier aux performances des protocoles de communication sans fil dans ces applications. Nous relevons le principal défi du WSN : prolonger la durée de vie du réseau autant que possible, tout en conservant des niveaux élevés de fiabilité et une faible latence. À cette fin, nous analysons la couche MAC et la couche de routage, et réalisons également des expériences au niveau de la couche physique. Sur la base de ces analyses, nous avons conçu de nouveaux protocoles MAC et de routage qui bénéficient du meilleur

de la technologie WuR. Dans cette thèse, nous présentons deux contributions principales : l'analyse à la couche MAC et la nouvelle solution à la couche de routage.

Dans notre première contribution, nous étudions les avantages, les inconvénients et les compromis de l'utilisation de la technologie WuR dans les réseaux WSN à multi-sauts en termes de taux de livraison des paquets, de latence et de consommation d'énergie au niveau de la couche MAC. Notre analyse est basée sur des évaluations utilisant COOJA [10], un simulateur pour les réseaux de nœuds ContikiOS [11]. Nos conclusions montrent qu'il existe un seuil dans la taille du réseau pour que le WuR fonctionne efficacement. L'augmentation de la taille du réseau au-delà de ce seuil dégrade considérablement les performances du WuR, ce qui fait d'un protocole MAC traditionnel à cycle obligatoire un meilleur choix pour une telle configuration. Nous montrons également que la reconnaissance du signal de réveil est problématique en présence de collisions car elle diminue sérieusement la fiabilité du réseau. Ensuite, nous réalisons une étude sur le comportement d'un prototype WuR lorsqu'il est soumis à un environnement bruyant réel. Nous analysons comment les interférences peuvent être considérées à tort comme des paquets valables et comment les traiter. Nous montrons l'importance de l'utilisation de la méthode Clear Channel Assessment (CCA) pour réduire les erreurs de transmission, ce qui se traduit par un taux de livraison de paquets plus élevé. En outre, nous extrayons certaines valeurs physiques clés du prototype qui peuvent servir à la modélisation des communications WuR. Enfin, nous fournissons une méthode permettant d'estimer la consommation de courant totale d'une application déployée sur un réseau basé sur le WuR. Les résultats montrent que les communications radio représentent une part négligeable de l'épuisement de l'énergie dans les scénarios de faible trafic, ce qui signifie que de nouvelles optimisations de la pile de protocoles de communication n'amélioreront pas la durée de vie des appareils finaux dans de tels cas.

La deuxième contribution va au-delà de la couche MAC pour analyser l'impact de l'utilisation de WuR à la couche de routage et explorer ses avantages. Nous y présentons LoBaPS, une approche inter-couches qui repose sur une structure de routage préétablie et combine le meilleur de deux mondes : l'efficacité énergétique et la fonction de mise en service permanente du WuR avec la stabilité d'une configuration bien connue du Routing Protocol for Low Power and Lossy Networks (RPL) [12]. De plus, nous mettons l'accent sur la redistribution des charges pour prolonger la durée de vie du réseau. Cette mesure, ainsi que la latence et le taux de livraison des paquets, ont été extraits de simulations. Les résultats montrent la robustesse de la solution car

le réseau peut s'adapter rapidement aux arrêts des nœuds. Encouragés par ces résultats, nous avons étendu le LoBaPS pour nous concentrer sur l'énergie dans une nouvelle solution appelée Energy LoBaPS (eLoBaPS). Les économies d'énergie réalisées se reflètent dans la durée de vie résultante qui est comparée à celle d'un protocole de référence basé sur WuR et LoBaPS. En outre, nous examinons le ratio de livraison de paquets du réseau dans le temps pour comparer la stabilité et la baisse finale des performances de fonctionnement, où l'eLoBaPS a un fonctionnement stable plus long et une baisse plus courte. En outre, nous présentons une nouvelle mesure appelée *productivité* qui reflète à la fois la fiabilité et la durée de vie quel que soit le scénario de trafic. En améliorant la répartition de la charge vers le cas idéal, notre protocole prolonge la durée de vie du réseau jusqu'à 77 En outre, le comportement du réseau devient plus stable au cours de sa durée de vie et la baisse avec la dégradation des performances est plus courte.

L'eLoBaPS présente de nombreux avantages mais nécessite une structure de routage pour fonctionner. Ses performances dépendent directement de la solution utilisée pour construire et maintenir la structure de routage Dans notre contribution finale, nous avons décidé de supprimer cette dépendance en concevant un protocole autonome appelé REFLOOD. Il s'agit d'un protocole de routage réactif qui exploite les caractéristiques du WuR. Nous comparons REFLOOD à une approche proactive via une série exhaustive de simulations réalisées dans ContikiOS/COOJA. Les résultats montrent que REFLOOD représente une amélioration de 300 Sa fonction de chemins multiples maximise les chances de réveiller une destination avec succès, améliorant le taux de livraison des paquets et réduisant la latence. En outre, nous avons montré qu'il est plus résistant aux changements de topologie. Enfin, l'absence de surcharge de contrôle dans les protocoles réactifs est particulièrement importante pour augmenter la durée de vie du réseau dans les applications à faible trafic avec des wake-up radios.

Structure de la thèse

Cette thèse est organisée en deux parties : La partie I pour l'analyse de la couche MAC et la partie II pour la solution de la couche de routage. Chaque partie contient un chapitre pour chaque contribution et un chapitre de conclusions avec un résumé de l'ensemble des contributions.

Le chapitre 2 décrit en détail la technologie radio de réveil, en fournissant les bases nécessaires pour suivre les prochains chapitres. À l'intérieur de la partie I, le chapitre 3 explore les avantages de la technologie WuR à la couche MAC, tandis que le chapitre 4 décrit les expériences que nous avons menées pour étudier le comportement de la technologie WuR dans un environnement

bruyant avec un prototype réel. À la fin de la partie I, nous présentons un résumé de l'analyse de la couche MAC. Passant à la partie II, le chapitre 5 décrit une introduction aux défis de l'utilisation de WuR du point de vue du routage et présente LoBaPS, une solution inter-couches axée sur la répartition des charges. Ensuite, dans le chapitre 6, nous présentons REFLOOD, un protocole de routage réactif autonome exploitant le WuR. Pour conclure la partie II, nous offrons un aperçu de ces contributions. Enfin, le chapitre 7 conclut cette recherche sur les protocoles de communication pour les technologies wake-up radio et propose quelques pistes de recherche potentielles.