Internet of highly mobile things

Cosmin Cobârzan

UNIVERSITÉ DE STRASBOURG

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

Laboratoire ICube – UMR 7357

THÈSE présentée par :

Cosmin COBÂRZAN

soutenue le : **29 septembre 2015**

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

Discipline/ Spécialité : Informatique

Internet of Highly Mobile Things

THÈSE dirigée par :

M. NOËL Thomas	Professeur, Université de Strasbourg

RAPPORTEURS :

M. TOUTAIN Laurent	Enseignant-chercheur, Télécom Bretagne
M. VALOIS Fabrice	Professeur, INSA Lyon

AUTRES MEMBRES DU JURY :

M. MONTAVONT Julien	Maître de conférence, Université de Strasbourg
M. GAYRAUD Thierry	Professeur, Université de Toulouse 3

UNIVERSITÉ DE STRASBOURG

Cosmin COBÂRZAN

Internet of Highly Mobile Things

Résumé

La mobilité devienne un partie intégrante de l'Internet des Object d'aujourd'hui, comme beaucoup d'applications (monitorage des animaux sauvage, suivi des cible dans le champs de bataille) sont impossible de mettre en œuvre juste avec des nœuds statiques. L'objective de cette thèse est de définir une nouvelle architecture de communication articule autour de la mobilité dans les réseaux avec pertes et a bas puissance (Low Power and Lossy Networks - LLNs) (réseaux des capteurs sans fils). Tout d'abord, nous avons analysé théoriquement l'auto configuration des adresses IPv6, fait avec toutes les optimisations disponibles dans Neighbor Discovery Optimization for IPv6 over 6LoWPAN. Cette étape est cruciale pour des protocoles qui donnent de support pour la mobilité dans des réseaux IP, comme MIPv6. Les résultats obtenues – taille des paquets trop grande et consumations énergétique importante pour les routeurs qui tournent Neighbor Discovery – n'ont amener a utiliser le IPv6 Routing Protocol for Low Power and Lossy Networks (RPL). RPL est développe d'el debout pour les LLN.

Notre deuxième contribution sont améliorer les opérations du RPL pour mieux supporter les nœuds mobiles. Enfin, nous avons développe une mécanisme inter-couche – Mobility Triggered-RPL – qui profite des actions dans le protocole avec préambule X-Machiavel à la couche accès au medium dans le protocole RPL à la couche routage.

Mobilité, Réseaux des capteurs sans fils, Dynamique du réseau, RPL

Résumé en anglais

Mobility is becoming an integrating part of todays Internet of Things, as many applications such as wildlife monitoring or target tracking in the battlefield cannot be done only with the help of static nodes. The goal of this thesis is to provide new communication architecture articulated around providing mobility support in Low Power and Lossy Networks (LLNs). First we analyzed from a theoretical point of view the IPv6 address auto-configuration with all optimizations made in Neighbor Discovery Optimization for IPv6 over 6LoWPAN. This step is of crucial importance for protocols that offer mobility support in IP networks, such as MIPv6. Our findings, increased message size that leads to fragmentation and high energy consumption for routers that are involved in Neighbor Discovery message exchange, have lead us to use the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL) in order to provide mobility support. RPL is build from ground up with respect to LLN requirements. Our second contribution enhanced RPL operations to support mobility management. Finally, we proposed a cross-layer protocol – Mobility Triggered-RPL – that leverages actions from the X-Machiavel preamble sampling MAC protocol into RPL.

Mobility, Wireless Sensor Networks, Network dynamics, RPL

UNIVERSITÉ DE STRASBOURG

École doctorale MSII

Internet of Highly Mobile Things

THÈSE

présentée pour obtenir le grade de

Docteur de l'Université de Strasbourg

(Mention SCIENCES - Specialité INFORMATIQUE)

par

Cosmin COBÂRZAN

Composition du jury

Directeur de thèse :	Prof. Thomas NOËL, Université de Strasbourg, France
Co-encadrant de thèse :	Dr. Julien MONTAVONT, Université de Strasbourg, France
Rapporteurs :	Dr. Laurent TOUTAIN, Télécom Bretagne, France Prof. Fabrice VALOIS, INSA Lyon, France
Examinateurs :	Prof. Thierry GAYRAUD, Université de Toulouse 3, France

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

To my family and fiancé.

Acknowledgments

I would like to thank assoc. prof. Julien Montavont and Prof. Thomas Noël for believing in my potential and taking a leap of faith enrolling me in the thesis. Their constant guidance helped me navigate the research world. Julien, I want to express my deepest gratitude towards all the efforts that you made during my thesis. It gave me strength to continue working beyond what I knew I was capable of.

I would also want to thank Prof. Fabrice Valois, Dr. Laurent Toutain and Prof. Thierry Gayraud for accepting to be part of my jury and review my thesis.

A big thank you for all my colleagues from Équipe Réseaux. I had a great time sharing the office with my fellow PhD colleagues Oana, Georgios, François and Julien. B. Thank you for all your unconditional support, all the productive discussion related to research and for all the activities after work that made me enjoy these three years spent in Strasbourg. I would also like to thank Antoine, Pascal, Stéphane, Pierre, Fabrice and Jean-Jaques for the fruitful conversation that we had and for all the insights that I got on France and french culture. Guillaume and Erkan were always there to help me with technicalities related to the FIT IoT-LAB experimental platform.

Special thanks to my Master adviser from the Technical University of Cluj-Napoca, assoc. prof. Emanuel Puşchiţã and to Ilie Nistor, as they were the first to see my potential and their gentle push toward research opened my eyes and made me enroll in this thesis.

During all the thesis I had constant support from my family and friends, without which it would have been much harder. Mom, dad, you always wanted what is best for me and you always pushed me further. Thank you for all your emotional support as I spread my wings and took to the sky to follow my dreams far away from home. My brother, Claudiu, along with his family, was also close to me all the way. He knew and understood better than any one the challenges of a PhD and always had the best encouragements for keeping me on track. I don't know how I may make it up to you :).

Last, but not least, I want to thank Diana, my fiancé. She offered me constant support and encouragement. I always felt she is close to me even though many times half the world separated us. You were the one that helped me keep my sanity and offered me a view of a bright future together.

Abstract

In recent years, new types of networks have emerged, enabling connections between sensors that have various communication capabilities and the Internet. This has lead to what we now call the Internet of Things. An important part of the Internet of Things are Wireless Sensor Networks (WSNs). The sensors communicate wirelessly in WSN, bringing new challenges to communication. (e.g. unstable links). Nevertheless, even though nodes move, associate or disassociate from the network, they need to remain reachable from other nodes. New applications developed for WSNs require mobile nodes (e.g. patient monitoring) which means that mobility support needs to be integrated in the network. The goal of this thesis is to propose new communication architectures articulated around mobility support in Wireless Sensor Networks.

Inside IP networks, mobility support is generally provided by protocols such as Mobile IPv6 (MIPv6). There are few proposals to use MIPv6 in a WSN environment and the corresponding results are not fully conclusive. In this context, our first contribution is to furthermore analyze the practicality of MIPv6 in a WSN environment. For this, we analyze the impact of IPv6 address auto-configuration with optimization of Neighbor Discovery for WSN on the performances of the WSN from a theoretical point of view. IPv6 address auto-configuration is a necessary step before MIPv6 operations can be performed for mobility management. Our analysis finds that the large message size needed by Neighbor Discovery (ND) and the energy consumption for routers that manage address auto-configuration questions the further integration of MIPv6 in WSNs. We have thus decided to focus our attention on IPv6 Routing Protocol for Low-power and Lossy Networks (RPL), which is specifically designed for WSNs. RPL has already a built-in mechanisms that manages the parent set, but it reacts slowly to changes in the topology and fails to prevent the disconnection of mobile nodes. Therefore, enhancements of RPL operations to support mobility management or using external unreachability mechanisms with RPL have been proposed.

Our second contribution is an enhancement of RPL operations for mobility management. We propose a reverse trickle algorithm for static nodes that act as preferred parents for the mobile node, alongside with advertisement of the mobility status from the mobile node. Our second contribution manages to decrease the control traffic overhead and the disconnection time of the mobile node from the DODAG and to increase the packet delivery ratio, when compared to other state of the art solutions. Even though we managed to reduce the disconnection time, the mobile node still has to actively track (i.e. synchronize itself with the received DIO messages) the connection with the preferred parent. We have thus continued to search for a better solution, where a mobility aware MAC layer provides support for RPL mobility management operations.

Finally, we propose a third contribution, that is a cross-layer mechanisms between X-Machiavel MAC protocol and RPL protocol, that improves RPL mobility management. The idea behind our main contribution is to trigger actions in RPL (e.g. change of preferred parent) when events related to mobile node transmission of data packets occur at the MAC layer (e.g. a static node acts as an opportunistic forwarder for the mobile node). Moving forward the layered structure of communication brings improvements over related unreachability detection mechanisms suggested in RPL. Our main contribution bounds the disconnection time of the mobile node close to the mobile node's sending data rate, improving along the packet delivery. The MAC and networking layer work now together which also determines a decrease in the control traffic overhead.

Publications

The contributions of this thesis have been published or submitted in several peer-reviewed international conferences and journals.

Journals

- C. Cobârzan, J. Montavont and T. Noël, *Experimental evaluation of mobility generated network dynamics in RPL*, **to appear in** Springer Journal of Internet Services and Applications: Thematic Series on the Internet of Things, 2015

International Conferences

- C. Cobârzan, J. Montavont and T. Noël, *Integrating mobility in RPL*, in proceedings of the 12th European Conference on Wireless Sensor Networks (EWSN 2015), Porto, Portugal, February 2015
- J. Montavont, C. Cobârzan and T. Noël, *Theoretical Analysis of IPv6 Stateless Address Autoconfiguration in Low-power and Lossy Wireless Networks*, in proceedings of the 11th IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF'15), Cantho, Vietnam, January 2015.
- C. Cobârzan, J. Montavont and T. Noël, *Analysis and performance evaluation of RPL under mobility*, in proceedings of the 19th IEEE Symposium on Computers and Communication (ISCC'14), Funchal, Portugal, June 2014.

Table of Contents

Chapter 1

Introduction and Context

1.1 Beginnings of Wireless Sensor Networks

We perceive the world around us through our senses which give us information about the world we live in. Taking inspiration from this, researchers and engineers translated the capability to sense to physical objects, called sensors. After having the first sensor, using more of them to perceive the environment on a broader area became a natural next step, along with interconnection between such sensors. As with many other technologies, the military invested first into sensor capabilities (like they have done for the first packet switching network, the Advanced Research Projects Agency Network (ARPANET)).

In the 1950s, the first distributed sensor network, Sound Surveillance System (SOSUS) [CK03], was deployed by the United States Army in the Atlantic and Pacific oceans to detect and track Soviet submarines. The network was composed of submerged acoustic sensors and is still in service today. Later, in the 1980s, the Defense Advanced Research Projects Agency (DARPA) started the Distributed Sensor Networks (DSN) project [DP10] to formally explore implementing WSNs, attracting along academia, through partners such as Carnegie Mellon University and the Massachusetts Institute of Technology Lincoln Labs.

These early deployments, expensive and bulky, focused on sensing and left aside the networking part, limiting the widespread of WSNs. Nevertheless, deployment costs came down over the years, as semiconductors became cheaper and miniaturization drove down sensor size. Increasing WSNs functionality is linked to advancements in four key areas: sensors, CMOS-based semiconductor devices, networking protocols and energy storage/generation technology. In this thesis, we will focus our attention on networking protocols and their inner workings. In the following, we will continue to present the structure of a Wireless Sensor Network and their communication stack.

1.2 The Structure of Wireless Sensor Networks and Low-power and Loosy Networks

Recent advances in wireless technologies together with the miniaturization of electronic devices have stimulated the appearance of new types of smart objects. Such objects, called sensors, are generally composed of:

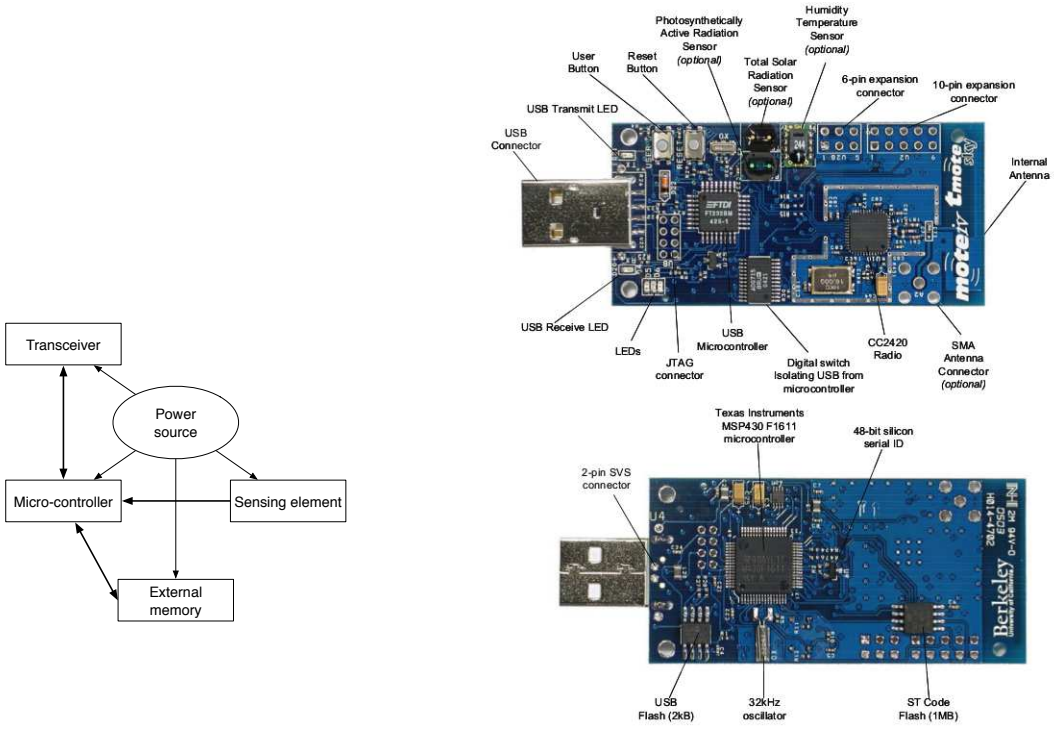
- a set of physical sensors (light, sound, motion, etc.) that constitute the sensing element
- a micro-controller which has a firmware (embedded software instructions) that controls the data collection and communication

– a transceiver which is used for wireless communications to send relevant data to a final destination in a multi-hop fashion (from one sensor node to another sensor node) or to receive information from other nodes

– an external memory that stores sensing information, either indefinitely or until the information is sent to the final destination

– a power source which ensures energy for all modules (i.e. sensing element, micro-controller and transceiver).

Figure 1.1a shows a generic diagram of the above mentioned modules, while Figure 1.1b presents the T-Mote Sky node, one of the most popular wireless sensor nodes, along with the location and identification of the major components which are present on the sensor node. As we can see, sensors can communicate between themselves (using the transceiver) and will form a new network class, called Low-power and Lossy Network.



(a) Generic diagram

(b) T-Mote Sky node

Figure 1.1: Sensor node

Interconnecting Low-power and Lossy Networks (LLNs) together with the Internet gave birth to what is now called the Internet of Things. By 2020, it is expected that more than 50 billion smart objects will be connected to the Internet [Iot], a number that will exceed by more than 8 times the predicted worldwide population at the very moment. However, networking smart objects together is a challenging task due to various constraints: objects have limited and non - replenishable energy resources, limited memory and limited computational power [Lew04]. Those constraints have led to new paradigms for connected objects since their requirements are different from the ones of regular computers connected to the Internet. This is especially true when it comes to computational resources as well as the available energy and communication range. The most envisioned applications for LLNs will only deal with low amounts of data while requiring energy efficiency in a context where communication range is of high concern. The available energy

limits the transmission power while the lossy nature of the wireless links add additional challenges to communication. Moreover, information flows towards a limited number of destinations when compared to other network types.

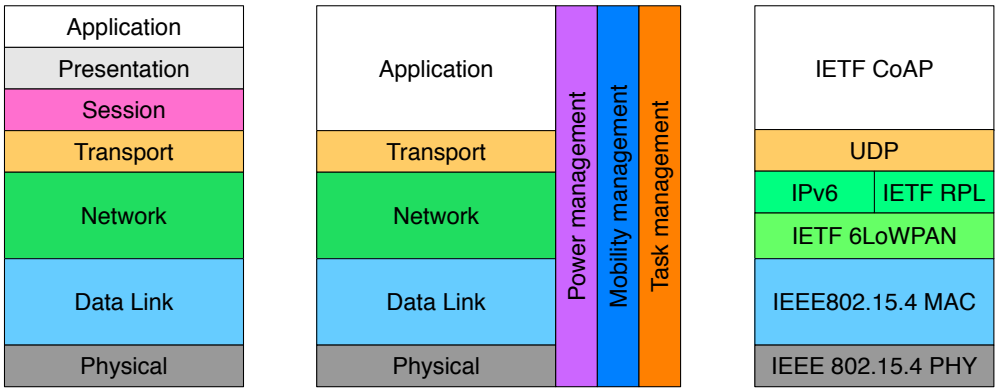
In the LLN family, connected objects, smart devices, constrained devices or sensors - which will be generically called *nodes* throughout this thesis - that communicate wirelessly, regardless of the wireless technology used, are grouped into Wireless Sensor Networks. Most communication is done in the Industrial, Scientific and Medical radio bands (ISM). Because of this, the standard Open Systems Interconnection (OSI) protocol stack from Figure 1.2a suffered changes to reflect WSNs requirements. Throughout this thesis, the terms LLN and WSN are used in an interchangeable way and refer to wireless nodes that form a network. In the following, we take a look at some of the similarities and differences of OSI and WSN protocol stacks, and then focus on the communication layers that are used as support for the proposed solutions thought this thesis.

Overview of similarities and differences between OSI and WSN protocol stacks. Communication between two hosts in an Internet Protocol (IP) network transits the communication layers as shown in Figure 1.2a. At the bottom, the *physical layer* ensures transmission of raw bit streams and is the physical interface of communication between hosts. Issues at this layer can be caused by mechanical, electrical or electromagnetic interferences as well as the distance between hosts. As it can be seen in Figure 1.2, the physical layer is present in both IP and WSN communication stacks. In the case of WSN, the physical links are wireless, made possible by using a radio chip that transforms electrical signals in electromagnetic waves.

Over the physical layer we find the *data link layer*. In the OSI stack, this layer provides the transfer of information between two adjacent hosts as well as frame-level error control and flow control. At this layer level, framing can become an issue when data has to be divided into chunks and header & trailer bits have to be appended. The data link layer in WSN provides different challenges: since the environment can be noisy and nodes might be mobile, the layer has to minimize packet collisions and has to be power/energy aware. The *networking layer* has the responsibility of path selection between end systems - the routing of data is done using different metrics in both protocol stacks. The routing path is in general made out of IP addresses (Internet Protocol version 4 (IPv4) or IPv6). This layer finds itself in both OSI and WSN communication stacks and provides the same role in both communication stacks. Over the network layer, the *transport layer* is in charge with end-to-end flow control and providing virtual end-to-end links between peer processes in the OSI protocol stack. In the WSN protocol stack, the flow control is also made by the transport layer. The main challenge here is to have a reliable communication. The *session* and *presentation* layers are present only in the OSI protocol stack. The session layer handles sessions between hosts by grouping several user-layer connections into a single session. At the presentation layer data can be encrypted, compressed or converted. The last layer, the *application layer*, provides the interface with any software application that needs communication capabilities in both protocol stacks.

The WSN protocol stack is build to serve sensing tasks, enable cooperative efforts of nodes and handle efficiently the limited energy budget on hand. In achieving these objectives, in the WSN protocol stack there are several planes that act across all layers: the *power management plane* shapes energy demanding tasks like transmission and reception; the *mobility management plane* detects and registers the node movement and helps layers keep the connectivity with a mobile node; the *task management plane* balances and schedules sensing tasks for nodes in a region. Management planes enable nodes to work together more efficiently and support goals like network lifetime increase.

As the community has build a solid body of research in recent years, many ideas are beginning to crystallize in WSNs. Moving on to the next step, that is standardization, the Internet Engineering Task Force (IETF) standardization organization leads the way in aggregating several elements from this body of research into standards. In Figure 1.2c, the generic layers from Figure 1.2b now have a concrete implementation. An additional sub-layer in the networking layer



(a) OSI protocol stack (b) WSN protocol stack [Aky+02] (c) Standardized WSN protocol stack [AP14]

Figure 1.2: Examples of communication protocol stacks

is added, namely the IPv6 over Low power WPAN (6LoWPAN) *adaptation layer*. 6LoWPAN is responsible for enabling constrained devices to support IPv6 protocol. IPv6 cannot be used as is in WSN due to the limited size of layer 2 frames in IEEE 802.15.4, therefore 6LoWPAN handles IPv6 header compression, fragmentation and reassembly. The network layer is divided into routing (using IPv6 Routing Protocol for Low-power and Lossy Networks (RPL)) and addressing (using IPv6). Transport layer functionalities are carried out by the User Datagram Protocol (UDP), which serves the application layer, that in turn uses the Constrained Application Protocol (CoAP).

1.3 Motivation

WSNs are deployed most commonly with static nodes (e.g. in Torre Aquila [Cer+09]) and have been shown to be relevant in numerous domains. Recently, WSNs have come to a merging point with the Internet, with a seamless interconnection between both worlds, that is now known as the Internet of Things.

Applications over WSNs continue to evolve, the latest trend is to move towards applications that leverage information from moving nodes (e.g. in patient monitoring [Dag+07], wildlife monitoring [All+07] or tactical support for the soldiers in the battlefield [Lee+09]). We can see that mobility is becoming an important factor, as ubiquitous connection to the network becomes the new norm. Managing nodes mobility is challenging, as the movement of nodes and their limited transmission capabilities determines changing the mobile node attachment point to the network. Nevertheless, such changes need to be done seamlessly and in a transparent way for applications running on the node, which until now has not been achieved.

The goal of this thesis is to provide efficient solutions to network mobility, in a way that ensures connectivity during the movement of a node in the network. The disconnection of a node is first signaled by the Medium Access Control (MAC) and networking layers, as they are the first ones to process received information. Keeping contact with a moving node is a challenging task, and up until now, solutions have focused on maintaining connectivity of communication from mobile nodes to the static nodes. Communication can be carried out bi-directionally (from a static to a mobile node and from a mobile to a static node), but exploring how connectivity can be maintained from static nodes to mobile nodes or from a mobile node to another mobile node is still under-explored.

Our focus to provide efficient mobility management solutions will be carried out with informations provided by the networking layer. Our choice of protocol at the networking layer to support node mobility can build connections bi-directionally which will thus allow to explore

mobility management for both communication directions.

1.4 Contributions

The goal of this thesis is to explore new communication architectures which are articulated around mobility support in WSNs. All proposed solutions are build with this goal in mind, bringing both enhancements to existing protocols and completely new solutions that provide mobility support. Mobility support can be achieved at the networking layer either by using protocols that manage the mobility using the IP address, such as Mobile IPv6 and the existing variations (e.g. Proxy Mobile IPv6, Hierarchical Mobile IPv6) or by using routing protocols, such as IPv6 Routing Protocol for Low-power and Lossy Networks, which is specifically designed for WSN.

First, we make an analysis of IPv6 address auto-configuration using the most recent optimizations proposed in the Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) standard from a theoretical point of view, in order to try to improve performances of Mobile IPv6 protocol.

IPv6 stateless address auto-configuration in WSN will provide necessary IPv6 addresses to all nodes in the WSN network. Having a valid IPv6 address enables nodes to communicate more efficiently (e.g. by using a routing protocol) and is an important step in protocols such as MIPv6. We analyzed optimizations brought to Neighbor Discovery protocol regarding message size and energy efficiency. This analysis will lead us to search for alternatives to achieve the thesis goal. We chose a routing protocol, RPL, specifically designed for WSN. RPL has a built-in mechanism that manages a node's parent set and adapts to slow changes in the topology, but this mechanism fails to prevent serious node disconnection from the graph due to mobility.

Second, we proposed an enhancement of RPL operations to support mobility management.

When a mobile node moves inside a WSN, from the routing perspective, specific actions must be taken to efficiently integrate the mobile node in the network. The communication between the mobile node and its preferred parent IN RPL will be done using a new message flow (e.g. a reverse trickle algorithm). Comparing our solution with other state of the art approaches through an extended simulation campaign, we obtain a lower control traffic overhead, a lower disconnection time of the mobile node from the graph and improve packet delivery ratios of the mobile node. Nevertheless, the mobile node must still track the connection to the preferred parent and synchronize with the received DIO messages. A better solution is to integrate a MAC layer that supports mobility with RPL operations, so that we can track closer the path of the mobile node.

Finally, we propose a cross-layer mechanism (between the MAC (X-Machiavel MAC protocol) and networking (RPL routing protocol) layers) that, through managing the unreachability of parents in RPL, enables RPL to support also mobility management.

RPL suggests using three unreachability detection mechanisms to manage the parent set (a pool of neighboring nodes that will give the next hop for communication from the mobile node). Through managing the parent set, a mobile node can find new attachment points in the network as movement causes disconnection from the former point of attachment. We implement the hints from lower layers via Layer 2 (L2) triggers, a generic solution, between X-Machiavel MAC protocol and RPL protocol. This led us to Mobility-Triggered RPL (MT-RPL), a cross-layer mechanism between these two aforementioned protocols. MT-RPL has been evaluated through both simulation and experiments, alongside two other unreachability detection mechanisms suggested by RPL. In MT-RPL, events from the MAC layer will be reported to RPL, which will take necessary actions to maintain the parent set up-to-date along with the connectivity of a mobile node to the network.

1.5 Structure of the Thesis

This manuscript is organized in six chapters. The first chapter introduces the Wireless Sensor Network architecture and presents the motivation and contribution of the thesis. In Chapter 2, we present solutions for mobility management at the link layer, highlighting characteristics of MAC protocols that enable seamless integration of a mobile node in the network at the link layer. We continue exploring mobility management solutions at the networking layer in Chapter 3. Here we present our first contribution, an analysis of IPv6 stateless address auto-configuration using the latest optimizations for WSN available for Neighbor Discovery protocol from a theoretical point of view. This analysis allows us to leave the MIPv6 path and move to alternatives such as RPL, a routing protocol that can manage mobility. In Chapter 4 we present our second contribution, an enhancement of RPL operations to support mobility. We analyze its performance through simulation, and show that we outperform two state of the art solutions. The mobile node has a better support for mobility than the built-in mechanism in RPL, but the mobile node has to actively track the connection with the preferred parent. That is why we continued our search to find new ways to provide seamless mobility support. In Chapter 5 we explore external unreachability detection mechanisms that can be integrated with RPL to provide mobility management. Our final contribution is based on a cross-layer mechanism between X-Machiavel MAC protocol and RPL routing protocol that brings improvements over the other analyzed solutions. The mobile node experiences lower disconnection time from the graph, as the disconnection from the graph is now bound to the data sending rate. Finally, Chapter 6 concludes the thesis with final remarks and openings of future perspectives.

Chapter 2

Link layer mobility support in WSN

2.1 Introduction to the WSN link layer

The link layer has a primary sublayer known as the MAC sublayer that provides channel access control mechanism, enabling several nodes to communicate in a network. Although in regular Ethernet networks we find in the link layer also the Logical Link Control (LLC) sublayer, this sublayer did not find its way in WSN. That is why, during all this manuscript, referring to the link layer or to the MAC layer means one and the same thing. The MAC layer provides support for unicast, multicast, anycast and broadcast communication services that one node may use during communication with other nodes in the network. It is also responsible for channel access policies, scheduling, buffer management and error control for communications between two adjacent nodes. In WSN, MAC protocols need to take into consideration energy efficiency, reliability and low access delay to transmission resources as priorities.

Communication between sensors in a WSN is done wirelessly and it has become very important to efficiently manage the limited resource at hand (e.g. available energy, limited transmission range). Available energy is the most constraining factor, as changing or recharging the sensor's battery is in many situations impractical. Thus, spending energy on powering the radio chipset in order to transmit data, which generally consumes the most energy [Du+10], must be kept at a minimum. One of the biggest means of reducing energy consumption in a WSN is through designing energy efficient MAC protocols. MAC protocols with wake/sleep periods are the most energy efficient MAC protocols. The radio chipset of a node is kept for as long as possible in sleep (low power sleep mode) and wakes up (turned on mode) just to transmit or receive packets. This is done periodically forming a cycle. *The duty cycle* is the ratio between the length of the listening period (when the radio chipset is on) and the total length of a cycle (*sleep time + wake up time*). This indicates how long a node is awake for transmitting or receiving data. With a low duty cycle, the node will spend less time in idle listening or overhearing. On the other hand, a high duty cycle can be used when the node has large quantity of data to transmit. Duty cycle introduces latency, as nodes cannot instantaneously send data when it is available and have to wait for the next time the radio is awake. Thus, a balance between the duty cycle and the latency of communication must be achieved. The concept of duty cycle introduces new constraints in WSN, as the sender and receiver in a communication must be in wake up mode simultaneously (i.e. synchronized), in order to successfully exchange data. The duty cycle is thus a new and relatively challenging task especially in multi-hop wireless communications, like the ones found in WSNs.

In recent years, researchers have also included in their protocol design data delivery efficiency (delay and throughput) alongside energy efficiency [Hua+13a]. Besides energy efficiency, a MAC protocol has to be scalable and adaptable to changes in the network [Dem+06]. *Mobility* is one of the changes in the network. We can define two types of mobility: *weak mobility* due to hardware

failure (e.g. battery depletion) or when a new node joins the network and *strong mobility* when nodes are integrated in devices that can move, like cars, or are carried by animals or humans and change their attachment point to the network [Ali+05]. Keeping a mobile node connected to a network at MAC layer is thus a challenging task, as it involves integrating the mobile node in the transmission schedule.

In the following, we first provide a general classification of the existing MAC protocols. Then, we will focus on proposals that are dedicated to support mobility at the MAC layer. Throughout our analysis, we will highlight strengths and drawbacks for each analyzed proposal. This analysis will help us choose the most appropriate MAC protocol that supports mobility and can provide meaningful information in cross-layer solutions (between the MAC layer and the networking layer), as is our main contribution described in Chapter 5.

2.2 General Classification of MAC protocols for WSN

The plethora of available MAC protocols can be classified in four categories: asynchronous, synchronous, frame-slotted and multichannel [Hua+13a].

Asynchronous protocols

This class of protocols uses either *sender initiated* transmissions, which reduces energy costs at the receiver or *receiver initiated* transmissions, which achieves higher throughput. Using asynchronous protocols, nodes do not pay the high price for keeping any synchronization with neighbors, as this is done before the transmission of each packet.

Sender initiated MAC protocols use in general Low-Power Listening (LPL), a common MAC layer technique to lower energy consumption [Sha+13]. Nodes periodically wake up and perform Clear Channel Assessment (CCA). CCA evaluates the energy level on the radio channel and compares it to a threshold. The channel is declared *free* if the energy level is below the threshold and *busy* if the energy is above the threshold. When CCA declares the channel busy, a transmission occurs on the channel. Thus, the node will stay awake and wait for packets, or if no packets are received, it will remain awake for a predefined time. Otherwise it goes back to sleep immediately. Nodes only know the duration of the wake up period, but not when the wake up period begins. Thus, a node sends a control packet called *preamble*, before the actual transmission. The preamble is longer than the sleep period of the node, ensuring that the receiver will sample the channel (perform CCA) during the preamble and wake up to receive the incoming packet. Between channel sampling intervals, if nodes do not detect any activities on the channel, they go to sleep achieving in this way a low duty cycle which improves energy efficiency.

X-MAC [Bue+06] modifies LPL to achieve low latencies and energy consumption. It breaks the long preamble into short strobes. The strobes contain the address of the intended destination, allowing nodes to decide whether to stay awake for the incoming data packet or to go back to sleep. X-MAC also adds pause time between strobes to allow the receiver to acknowledge the strobe and end early the preamble period, as seen in Figure 2.1. Non-receiving neighboring nodes that overhear the transmission can go back to sleep after receiving the first strobe, if the advertised target address belongs to another node.

Receiver initiated MAC protocols employ a different strategy than sender initiated MAC protocols when a data packet must be sent. Nodes wake up and send poll packets in the neighborhood, signaling that they are ready to receive data packets from sending nodes. Nodes with data to send will respond to the polling packet by sending their data. Here, the main problem is deciding how to poll neighbors for data packets. The rate of polling influences network performances: too slow will lower throughput and increase average delay while too quickly will lower performances due to collisions with other polling packets. After polling, if no data is received, the node goes back to sleep. The Low Power Probing (LPP) mechanism [ME+08], allows each node to periodically poll a probe packet to neighbors

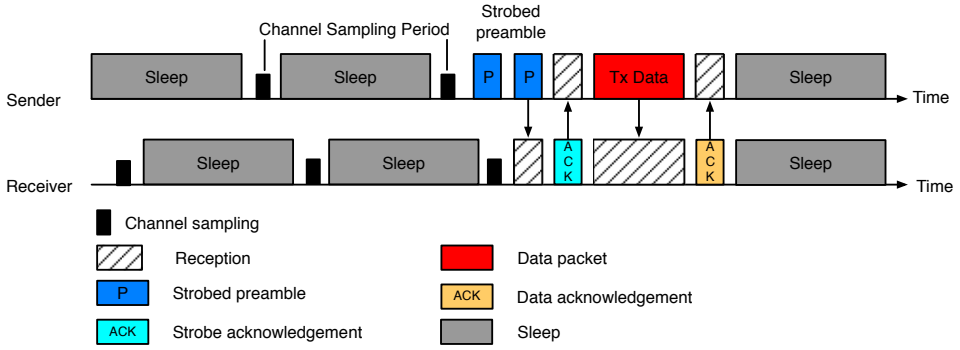


Figure 2.1: X-MAC protocol operation

requesting an acknowledgment. In LPP, if the acknowledgement is received, the node remains active and starts waking up other nodes by acknowledging their probes; otherwise, it goes back to sleep. Data packets are sent after all nodes are awake.

Authors of RI-MAC [Sun+08], extend LPP. In RI-MAC, the data packets are sent immediately after an acknowledgement for the poll packet (called beacon in RI-MAC) is received. A node wakes up periodically and broadcasts a beacon, signaling that it is ready to accept data packets. If there is a node with pending data (the Sender in Figure 2.2), it listens continuously on the medium for beacons from the intended receiver (the Receiver on Figure 2.2). The sender starts sending the data immediately after receiving the beacon from the receiver. After the data is successfully received, the receiver will acknowledge it by sending another beacon. If the sender sends all available data, it will go back to sleep.

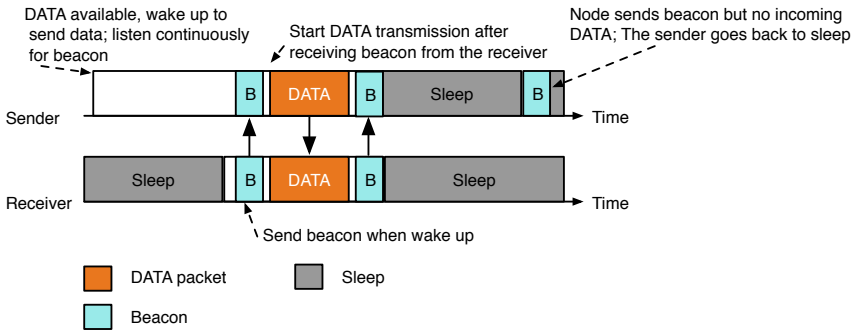


Figure 2.2: RI-MAC protocol operation

Asynchronous protocols have *advantages* both in sender initiated transmissions - where energy can be saved by both ends of communication as well as in receiver initiated transmissions - where the channel utilization is minimized as poll packets are used to trigger transmissions. *Disadvantages* of asynchronous protocols, regardless of transmission mode, come from the different sleep-wake up periods of nodes, which makes packet broadcast problematic.

Synchronous protocols

Aligning the active time of nodes ensures reliable communication between them. However, in order to achieve this, we have to pay the cost of synchronizing these active periods, which introduces communication overhead. The synchronization can be done using a schedule. The schedule specifies how long a node should sleep, in order to conserve energy, and when a node should wake up, in order to participate in communication. Nodes listen to the medium in their neighborhood for schedules. If the node does not hear any schedule from

other nodes, it will broadcast its schedule and will become a synchronizer. If other nodes in the neighborhood receive the schedule and do not already have one, they will follow the received schedule. The schedule can be propagated several hops and all nodes that follow the same schedule form a *cluster*. Depending on the topology, a node can receive two schedules and may choose to follow both, waking up for communication according to each schedule. These nodes, with two schedules, can become bridges between clusters. In synchronous protocols, all nodes in a cluster wake up simultaneously and share the common active period. During the common active period, nodes must contend for the channel in order to send their data.

S-MAC [Wei+02] is a classical synchronous protocol, where all nodes that follow a schedule form a cluster. Clusters share fixed time schedules and use local time synchronization (see Figure 2.3). The schedule is divided into three periods: SYNC, DATA and Sleep. All nodes in the same cluster will wake up at the beginning of the SYNC period in order to synchronize their clocks with each other (Sender 1 and Sender 2 are synchronized to the Receiver in Figure 2.3). During the SYNC period, a node sends a SYNC message, containing the node ID and the next wake up time (Sender 1 also has its own schedule in Figure 2.3). This message constitutes an overhead to communication, but it allows other nodes to synchronize with the next wake up time of the node. Then, in the DATA period, nodes with data packets contend for channel (only Sender 2 has data to send in Figure 2.3). Each transmitted packet has a duration field that indicates how long the remaining transmission will last. This value is recorder by nodes that receive the packet, is checked periodically and is decremented at each check, until the value reaches 0 (this is called virtual carrier sense). Then, a CCA is performed on the physical layer. The medium is determined to be free when the virtual carrier sense is 0 and the CCA returns an idle channel. Nodes send Request to Send (RTS) messages towards the receiver when the medium is free, and wait Clear to Send (CTS) messages from the receiver. If a CTS message is received, the sender will continue and send the data packets to the receiver. Nodes not involved in data packet exchange or that do not receive a RTS message, go back to sleep quicker. Thus, each cycle a packet can be forwarded only one hop. SYNC messages are sent periodically to allow new nodes to join an existing neighborhood. Initially, nodes should stay awake long enough to learn and follow an existing schedule before choosing an independent one.

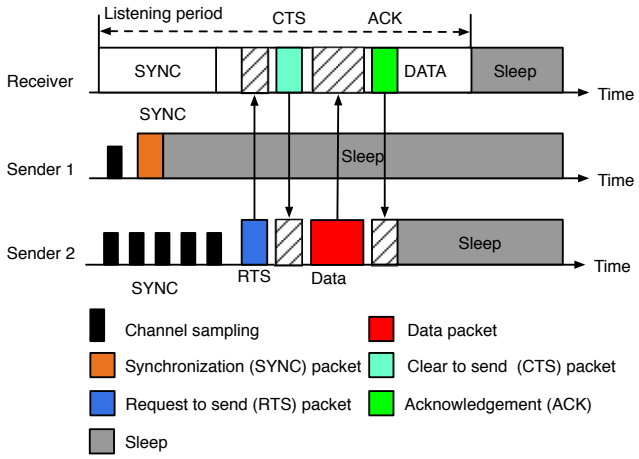


Figure 2.3: S-MAC timing relationship between a receiver and different senders [Wei+02]

Synchronous protocols have as *advantage* the fact that all nodes wake up at the same time which guarantees communication between nodes once the channel is acquired. On the other

hand, a *disadvantage* of synchronous protocols is represented by the constant overhead in communication in order to maintain nodes synchronization and that a dedicated time for sending and receiving packets is not guaranteed for any node (nodes still need to contend for the channel), even though the nodes are synchronized.

Frame-slotted protocols

Frame-slotted protocols organize communication using Time Division Multiple Access (TDMA). TDMA is used with stricter global time synchronization than in synchronous protocols, ensuring collision-free data transmission, as nodes have dedicated time slots in which they transmit. These protocols are generally favored in small-scale networks such as Wireless Body Area Networks (WBANs) [Hus+10], as interference between nodes with other types of protocols can cause collisions and such applications need increased reliability.

One of the most known frame-slotted MAC protocols for WSN is IEEE 802.15.4 MAC protocol in beacon-enabled configuration [802a]. IEEE 802.15.4 introduces the concept of superframe (see Figure 2.4). Nodes that control the superframe, which are called coordinators, periodically send a control packet named beacon that contains control information. The beacon allows nodes to enter and participate in the superframe. Two successive beacon packets delimit the superframe (with a beacon interval periodicity). The superframe is divided into three parts: the Contention Access Period (CAP), Guaranteed Time Slot (GTS) frames and an inactive period. During the CAP, nodes request GTS slots (which can be one or mode time slot long) for transmission. The access in CAP is done using a slotted Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) solution. Nodes go back to sleep once they participated in the superframe, until a next beacon is received or transmitted. The period of the CAP and all GTS slots form the superframe duration.

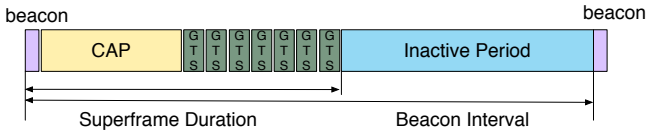


Figure 2.4: Superframe structure of IEEE 802.15.4 MAC in beacon-enabled configuration [Iov14]

The GTS slots are generally reserved for real-time services. The slot reservation technique is similar to TDMA, but GTS slots do not have necessarily a fixed length (they are adjusted by parameters in the beacon message). Nodes willing to reserve a GTS slot need to send a request to the coordinator. The coordinator allocates GTS slots on a first come, first serve basis [802a] and sends an acknowledgement to the requesting node. The next beacon will have a GTS descriptor (with a start index of GTS slot and the number of allocated GTS slots for the requesting node). All these operations are illustrated in Figure 2.5. During a GTS, nodes that exchange data (as indicated in the beacon) will send their data when the GTS frame starts, without using CSMA/CA, if the data size fits in the allocated GTS frames.

Frame-slotted protocols have as *advantage* an increase of the throughput between nodes, compared to synchronous protocols. One *disadvantage* can be the additional overhead introduced to allocate slots and maintain synchronization. In addition, nodes that loose synchronization loose also their allocated transmission slots.

Multichannel protocols

Parallel data transmission begins to attract attention, as WSN begin to incorporate multichannel capabilities. Radio bandwidth is limited in WSN, so it is desirable to develop multichannel protocols that can handle bursty traffic and provide multi-task support [Hua+13b].

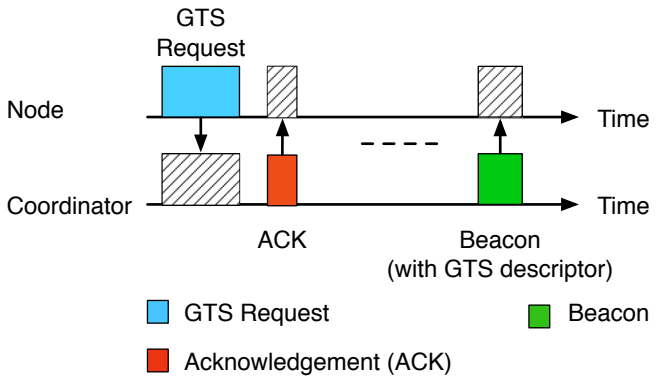


Figure 2.5: GTS request

The main concerns in multichannel protocols are channel allocation and cross-channel communication in order to achieve high energy efficiency and lower cost. As expected, multichannel MAC protocols designed for general wireless networks are not usable in WSNs if we consider two facts:

1. there is a need for more powerful radios for multichannel sensing
2. due to the small data packet size in WSN (127 bytes), the RTS/CTS control packets sent in standard IP networks (e.g. in IEEE 802.11 [802b]), no longer constitute a small overhead that can be ignored in WSNs [Zho+06].

That is why dedicated solutions for WSN have to be considered.

One of the dedicated solutions in WSN with multi-channel support is Multi-Channel Lightweight MAC - MC-LMAC [Inc+11], a schedule-based multi-channel MAC protocol. The protocol takes advantage of contention and collision-free parallel transmissions on different channels. MC-LMAC combines channel with timeslot assignment in a distributed way that guarantees collision-free transmissions on timeslot/channel pairs. A timeslot has two parts (shown also in Figure 2.6):

- a common frequency phase (CFP), where nodes switch to a common control channel where they learn if any transmission is due to them or can place requests to send data to a destination.
- a split phase (SP), in which senders and receivers switch to the channel on which the data transmission will take place. This means parallel transmission on different channels.

Several timeslots are grouped in a frame.

Before sending or receiving any packets, nodes should first synchronize with the network in order to send and receive messages with correct timing. Synchronization is done using a hierarchical structure, where the parent of nodes (the parent is the closest neighbor of a node with regard to the minimum hops until the sink) sends information about slot and frame numbers. Each slot is associated with a channel (on a given frequency), meaning that nodes that own the slot will transmit data, after the CFP for the timeslot, on the respective channel. The number of slots available in the CFP represents the number of channels available for parallel transmissions (e.g. three channels in Figure 2.6). Nodes in the synchronization phase must select a free slot in the common frequency part, in order to reserve the associated channel for their data transmission. Nodes that own a timeslot advertise in this slot (during CFP) the node to which they want to send data. After the CFP, nodes switch their radio to the associated channel (sender and receiver), entering the split phase. In the split phase, the sender first sends a control message (CM), which

can be considered a preamble, and than continue with the data packet. If the data packet is successfully received, the receiver will send an acknowledgement to the sender. This acknowledgement is sent on the channel owned by the receiver, in the control message part. Thus, the sender must switch its radio to the channel of the receiver to listen for acknowledgements for its data packet in the next slot.

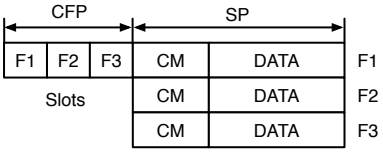


Figure 2.6: MC-LMAC timeslot structure [Hua+13b]

The main *advantage* of multi-channel protocols is the ability to have multiple parallel transmissions, increasing the capacity of the WSN. The main *disadvantage* is the high overhead for communication.

Table 2.1 summerizes the main advantages as well as the disadvantages of the four types of protocols that we have previously mentioned.

Protocol type	Advantage(s)	Disadvantage(s)
Asynchronous	Energy can be saved at both ends of communication. Channel utilization is minimized when poll packets are used.	Different sleep-wake-up periods. Problematic broadcast.
Synchronous	Nodes follow a schedule, and so they are always synchronized. After a node acquires the channel, data transmission is assured.	Constant overhead to maintain nodes synchronized. Nodes must contend for channel in order to transmit their packet.
Frame-slotted	Guarantees transmission resources between nodes (no more contention for channel). Increase of throughput when compared to synchronous protocols.	Increased latency in communication.
Multichannel	Parallel transmissions on different channels without collisions or contention.	High communication overhead.

Table 2.1: MAC protocols in WSN

Comparison of general MAC protocols in WSN From Table 2.1 we can see that there are several solutions available in the literature. Each type of protocol respond to specific problems, which were raised by different scenarios. Depending on the needs, we have asynchronous protocols that are focused very much on energy efficiency and minimization of channel usage. We then have synchronized protocols which ensured network wide communication between nodes, as a common sleep-wake-up schedule guarantees to all nodes the availability of remote peers, unlike the asynchronous protocols, where for each packet synchronization must be done. Still, nodes must contend for the channel before sending any data, so communication is not guaranteed, even though all nodes are awake at the same time. Frame-slotted protocols try to solve this last problem by providing slots of time when communication is guaranteed and no contention to the channel is present. This contributes to an increase of throughput when compared to synchronous protocols. Still, in low traffic conditions, the frame-slotted protocol wastes transmission resources if nodes do not have data to transmit, leading to low channel utilization. Lastly, multichannel protocols are a response to the limited radio bandwidth available in WSN and try to parallelize

transmissions. This achievement brings along collision and contention free communication for multiple nodes in the same time using different channel frequencies.

In this section, we made an overview of the major classes of MAC protocols available for WSN and have seen their strengths as well as their weaknesses. In the next section, we make an overview of mobility-oriented MAC protocols using the same classification of protocol types.

2.3 MAC protocols with mobility support in WSN

We have mentioned in the introduction of this chapter two types of mobility, weak mobility and strong mobility, which now will be explained further.

Weak mobility [Ali+05] occurs when changes in the network are seldom: nodes deplete their batteries or join the network. MAC protocols are generally suited to deal with this type of mobility and can adjust accordingly. If we look at S-MAC, we have seen that nodes update their knowledge about surrounding nodes when they exchange synchronization messages. X-MAC sends preambles long enough so that receiver nodes will be awake and acknowledge it. In RI-MAC, nodes send periodically beacons when they are available to receive data packets. Change is however known only at the beginning of the active period, leading to delays in packet transmission when changes occur in the topology. If we look at the frequency of this changes, which seldom occur, the delay can be tolerable and will be short lived [DD13].

On the other hand, strong mobility is mainly characterized by physical movement of nodes, determined by deliberate movement of nodes by humans or by external forces like wind or water. Many applications can be developed around strong mobility: patient monitoring [Dag+07] or nurse tracking in a hospital [Che+09], disaster recovery workers on site [Lor+04], sensors in oil extraction and refinery [Cha+08] that help avoid dangerous situations, or sensors that provide tactical support to soldiers on the battlefield [Lee+09]. In a strong mobility scenario, frequent topology changes introduce several problems:

- Data transmission is more vulnerable to failure, as mobility can deteriorate the link quality.
- Transmission delay of data packets from a mobile node when it enters a network, or when it changes frequently its attachment point, as establishing connectivity with a new attachment point may be time consuming.
- Depending on the type of MAC protocol, when mobility is present, packet collision increases for contention-based solutions (as more nodes contend for the channel), while for scheduled-based solution, as nodes change their point of attachment, two-hop neighbor information and schedules is likely to become inconsistent.

We will continue to present mobility-oriented solutions at MAC layer that try to solve the above mentioned problems. We will follow the same classification of protocols that was introduced in the previous section.

2.3.1 Asynchronous protocols

With asynchronous protocols, the inclusion of a mobile node in the network is done in a simple way. When data needs to be sent, the mobile node will evaluate if the channel is clear (using CCA) and send a preamble long enough so that the receiver wakes up and receives the data packet.

MA-MAC [ZD10], the lightweight mobility-aware medium access control protocol, is an extended version of X-MAC [Bue+06]. In static scenarios, MA-MAC operates like X-MAC, but when mobility is detected, MA-MAC tries to make a seamless handover to a new attachment point. In MA-MAC a node may find itself in five states: sleep, receive, send, discover and handover. When the node is powered up, it enters in sleep state. Once data is available to be transmitted, the node goes to send state. A node with mobility capabilities has defined two

thresholds in MA-MAC to trigger mobility related operations. The thresholds are based on Received Signal Strength Indicator (RSSI) in received ACK packets, which is then transformed in distance [PAP08].

The first threshold signals to the mobile node that seamless handover must be initiated, while the second threshold marks the upper limit regarding the distance a mobile node has traveled before it needs to find a new attachment point. After the first threshold is exceeded, the mobile node actively searches for a new attachment point by broadcasting its data packet with handover requests piggybacked. As packets are broadcasted, the current receiver of packets from the mobile node will not send acknowledgments anymore to the mobile node, letting new potential static nodes to which the mobile node can attach send acknowledgments for packets sent by the mobile node. If any broadcasted packet is acknowledged before the second threshold is exceeded, the mobile node enters a handover state and resumes data transmission only with the new attachment point. Nodes that overhear the broadcasted packets from the mobile node, but are not chosen as new attachment points, go back to sleep.

One *advantage* of MA-MAC is that the need of a handover is known in advance, with the help of the two thresholds. However there are also *disadvantages* for MA-MAC as the estimation technique for distance, based on RSSI, is inaccurate in real implementation and leads to performance under-achievements [CT10]. In addition, network density plays an important role for the mobile node, as it needs constant coverage from other static nodes along the path it has, in case a handover is due. Once the first threshold is exceeded, other nodes in the neighborhood must be awake during the broadcast of packets for a successful handover, otherwise the mobile node will become disconnected. Also, multiple nodes can receive the broadcasted packet and send simultaneously the acknowledgement, increasing the power consumption or nodes.

Another solution that extends X-MAC operations and enables communication for mobile nodes in the network is MoX-MAC [Ba+11]. Here, authors advocate that the mobile node should not transmit preambles if it detects other transmissions in its neighborhood. In addition, the mobile node should only send its data to static nodes in the network. Once the mobile node overhears transmissions between other static nodes, it will listen to the medium for acknowledgements of preambles (those are sent by static nodes that receive the preamble and acknowledge it). The mobile node, as it receives the acknowledgement, prepares a data packet destined to the source of the preambles (the destination in the received acknowledgment). The mobile node will choose a backoff period, to allow the static nodes to exchange packets, after which it will send its data packet to the node that sent the preamble (the node which sent already its data packet). Authors introduce this backoff period to mitigate potential collisions between packets from mobile and static nodes. Static nodes, once they sent their data, wait for a short time, unspecified by the authors, for any packet from mobile nodes. On the other hand, if there is no ongoing transmission in its neighborhood, the mobile node will use regular X-MAC operations and sends a preamble to find static nodes.

MoX-MAC has as *advantage* the simple integration of mobile nodes in the network, regardless of density. The mobile node, once it overhears an acknowledgement from a static node, can try to send its data packets toward the destination of the acknowledgement, after a backoff period. Still, there are some *disadvantages* when it comes to delay in communication: mobile nodes must wait for transmissions between static nodes before sending their data packets. Also, the mobile node can receive the acknowledgement from the destination of the preamble, but the mobile node has no guarantees that it can reach the sender of the preamble.

Based on X-MAC operations, but allowing the mobile node to steal the communication channel from other static nodes and to opportunistically forward its data packets, X-Machiavel [Kun+13] is a mobility oriented preamble sampling MAC protocol. X-Machiavel works on the premise of both static and mobile node cohabitation in the same network. Thus, X-Machiavel will change X-MAC behavior to give mobile nodes a head start for data packet transmission.

When the channel is idle, packets from the mobile node can be opportunistically forwarded by static nodes to the final destination. When the channel is busy, as the mobile node can overhear ongoing transmissions of other static nodes, it is able to steal the channel from them and send its own data first, before the static node that sent the preamble. These operations are possible

as X-Machiavel adds two new fields in the packet header:

- In the *type* field, a packet will be identified as being: a preamble frame (type P0, P1 or P2), a data packet (type DATA), an acknowledgment for a preamble (type PK0 or PK1) or an acknowledgment for a data packet (type Acknowledgment (ACK)). Preamble strobes of type P0 are used by the mobile nodes and signal that other mobile nodes cannot steal the channel. It also notifies static nodes that they can involve themselves as opportunistic forwarders and accept the pending data from the mobile node on behalf of the final destination. P1 type preamble strobes are sent by static nodes and advertise that they allow channel stealing by mobile nodes. Lastly, preamble strobes of type P2, sent from the static nodes, grant their data transmission as no node can steal the channel anymore. Acknowledging P0 preambles by static nodes is done with PK0 acknowledgment, when a P0 preamble is received but it is not destined to them. When a preamble is destined to the static node, they will acknowledge it with a PK1 type acknowledgment.

- In the *flag* fields, the mobile nodes will set on the most significant bit (MSB) a M flag in each sent data packet. This will allow a static node that receives data with the M flag to forward the packet with priority towards the destination. P2 preambles will be used to prohibit channel stealing by other nodes when data from the mobile nodes transits the network.

X-Machiavel can operate during all states of the medium (idle of busy), always allowing a mobile node to access with priority the transmission resources over static nodes. When the channel is idle, a mobile node sends P0 preambles, with the destination of the remote peer set in each strobe. If the remote peer is in the neighborhood, X-MAC principles apply: the peer will acknowledge the preamble with a PK1 acknowledgement, after which the mobile node will send the data packet and wait for the acknowledgement from the remote peer.

However, as the mobile node moves, it is likely that the intended destination of the P0 preamble from the mobile node is not in the neighborhood, but another node receives the P0 preambles from the mobile node. This static node now acts as an opportunistic forwarder and is ready to receive data from a mobile node (see Figure 2.7). The forwarder waits for a back-off period of $T_w = random(T_p/2, T_p)$, with T_p being the time between two consecutive strobes, before acknowledging the preamble with a PK0 preamble. This allows the remote peer, if it is in the neighborhood, to send first a PK1 acknowledgement. The forwarder, once it receives the data from the mobile node, will continue to relay it using P2 preambles, until the remote peer is reached.

It may happen that the mobile node finds itself in an area where congestion prevents it to send any data. Nevertheless, the mobile node can take over the medium from static nodes, by sending its data packet between two strobed preambles of the static node (two P1 preambles in Figure 2.8). The mobile nodes waits T_w, just as the static node does in the previous case, which allows the receiver to acknowledge the preamble and if no acknowledgement is sent on the medium, the mobile node sends its data packet. The static node will receive the data and will forward it using P2 preambles. The mobile node waits for the P2 preamble, which now acts as an acknowledgement from the static node for successful data reception. After the static node sends the data from the mobile node, it still has its own data to send (as it has advertised a P1 preamble), so it will continue advertising P2 preambles, in order to prevent channel stealing. Using this technique, the forwarder will manage to send also its data before another transmission from the mobile node.

X-Machiavel has *advantages* as, at any time, the mobile node is able to send its data packets, either by steeling the channel from static nodes or by using an opportunistic forwarder. Static nodes in the network give support to the mobile node and facilitate its transmissions. In addition, with X-Machiavel, a mobile node entering in the network does not disturb network operations. Whether it tries to steel the channel or to use an opportunistic forwarder, the mobile node backs-off, to allow the original receiver to capture the channel, and only if the receiver does not

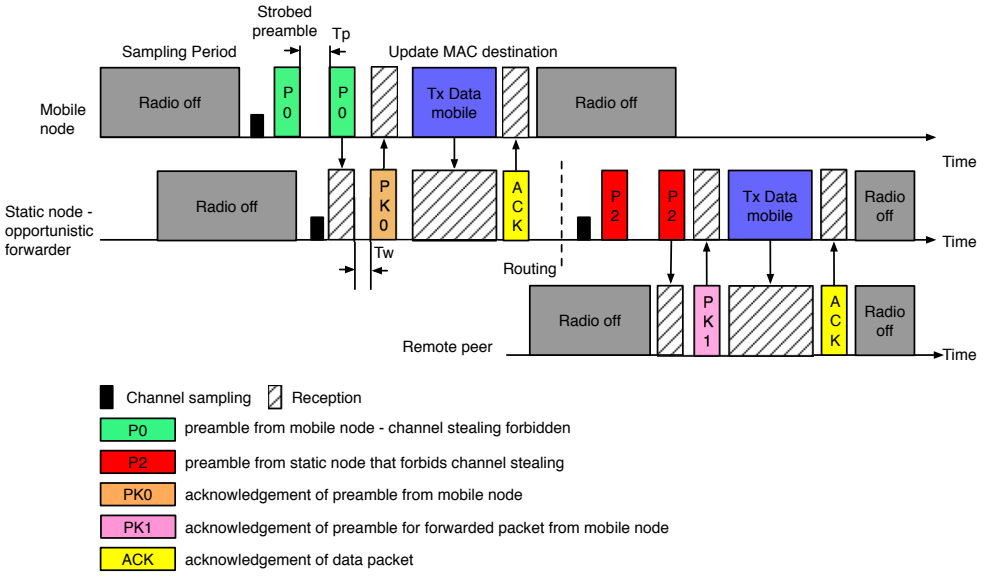


Figure 2.7: Opportunistic forwarding in X-Machiavel

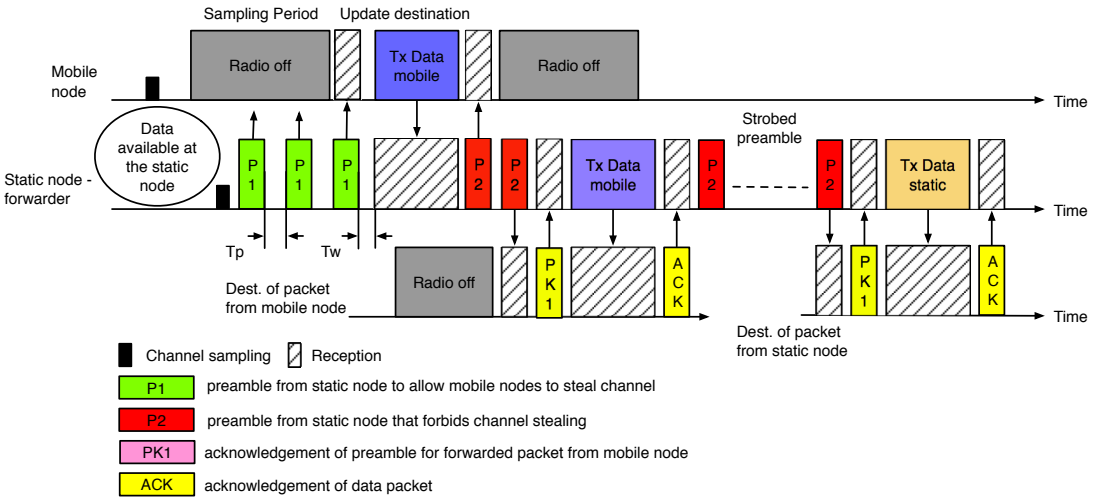


Figure 2.8: X-Machiavel channel stealing

reply before the end of the backoff, the mobile node will use the channel. As *disadvantages*, X-Machiavel does not present fairness between node types (mobile nodes have priority over static ones). Broadcasting packets is still problematic, just like in X-MAC.

2.3.2 Synchronous protocols

Synchronous protocols allow nodes to listen for the entire synchronization period for schedules from other nodes, in order to discover nodes in the neighborhood (neighbor discovery at MAC layer). This procedure is done periodically. Depending on the number of discovered parents, the node can increase or decrease the periodicity of neighbor discovery. Mobile nodes also perform neighbor discovery, so that along the path they take, they can synchronize to other nodes and exchange data.

Based on S-MAC [Wei+02], the mobility aware MAC protocol for sensor networks (MS-MAC) [PJ04] is one of the first protocols to extend S-MAC support of mobile nodes. MS-MAC considers that there are static as well as mobile nodes in the network. When a mobile node enters the network, the surrounding static nodes inside a predefined range, R, form an *active space*. Inside this active space, neighbor discovery will be performed more frequently then in the rest of the static network, which allows monitoring the mobile node. Nodes operating with MS-MAC are aware of mobility in their neighborhood based on RSSI levels from received SYNC (SYNC) messages (during the neighbor discovery period). Thus, a mobile node will be able to stay awake more time to ensure the synchronization with a new cluster before previous neighbors become unreachable [DD13]. Intra-cluster mobility can be handled by S-MAC, as the schedule does not change inside the cluster. Nevertheless, when inter-cluster mobility is detected, as nodes change their virtual cluster, mobile nodes will be disconnected from the network [JK07].

One *advantage* of MS-MAC is that the mobile node can continue exchanging data packets with a neighbor while synchronizing with a new virtual cluster. However, MS-MAC has a great *disadvantage* when it comes to energy efficiency. Frequent neighbor discovery periods lead to high energy consumption for both the mobile node and static nodes around it. The protocol trades low latency in setting up a connection with a virtual cluster for the mobile node with higher energy consumption of all nodes inside the active area. In addition, using RSSI to measure the distance is inaccurate and drops the performance of the protocol (in [CT10] authors showed that received RSSI values have a non-linear decrease rate and introduce errors in distance estimation).

Authors of Mobility Dynamic Sensor MAC, MD-SMAC [Ham+09] propose a mobility aware delay sensitive MAC protocol for WSN that tries to improve MS-MAC in the energy efficiency problem. Regarding mobility, MD-SMAC extends MS-MAC in three areas:

- In MS-MAC, all nodes in the active space, once mobility is detected, perform neighbor discovery. With MD-SMAC, only the mobile node keeps the shorter neighbor discovery period for synchronization, gathering schedules of nodes as it moves.
- MS-MAC imposes a higher neighbor discovery period for all nodes in the active space, with no mechanism to return to the previous neighbor discovery period once the static nodes are outside the active space. MD-SMAC leaves the burden of higher neighbor discovery periods only to the mobile node, the static node retaining their periodicity for neighbor discovery. After the mobile node acquires a new schedule, the periodicity of neighbor discovery during synchronization returns to values before mobility was detected (when the mobile node is inside a virtual cluster). This will contribute to energy savings at the mobile node.
- In MS-MAC, after the mobile node acquires a new schedule, as it enters a new virtual cluster, the mobile node will still keep the old schedule from the cluster that it has just left. This increases the energy consumption of the mobile node, as now the mobile node will follow two schedules. MD-SMAC drops the old schedule once the mobile node acquires a new schedule as it enters a new virtual cluster.

MD-SMAC keeps MS-MAC operations for mobile node, but reduces energy wastage mainly in the static part of the network (i.e. let static nodes have a higher neighbor discovery period), turn-

ing a disadvantage of MS-MAC into an *advantage*. MD-SMAC shares some of the *disadvantages* of MD-SMAC, namely the reliance on RSSI measurements to determine nodes mobility.

Another solution for mobility management based on S-MAC is AM-MAC [Cho+08], an adaptive mobility-supporting MAC protocol. AM-MAC is a random-access based MAC protocol for mobile sensor networks. The scheduling adaptation does not rely anymore on periodic neighbor discovery (as we have defined it at the beginning of the section), but on information from border nodes, providing thus fast and energy efficient scheduling adaptation. The border nodes operate between virtual clusters, following all schedules and relaying packets between clusters. The mobile node will receive SYNC messages from the border node once it nears the edge of the virtual cluster. The SYNC messages identify a border node with a flag and containing schedules that the border node follows.

Mobile nodes have an energy efficient secondary listening period. As a mobile node approaches a border node, it will receive SYNC packets with a border router indicator. This signals to the mobile node that it may soon enter a new virtual cluster. The mobile node will thus begin to listen also to any other schedules that it receives in the SYNC packet from the border node. The new schedules that it listens to are considered secondary schedules, as the schedule from its own virtual cluster is considered the primary one. The listening of the secondary schedule is done only for the duration when SYNC messages are expected (the SYNC period). The mobile node will not necessarily remain awake during the data exchange part. This saves energy at the mobile node.

Now that the mobile node has both primary and secondary schedules, it has to perform a smart schedule adaptation. While moving away from the border node, the mobile node will receive SYNC messages that will match only one of the schedules that it follows. When no more SYNC messages are received from the border node, the mobile node will continue to follow only the schedule for which it receives consistent SYNC messages.

AM-MAC has as *advantage* a seamless handover, as mobile node choose the schedule according to the received SYNC message periodicity. However, AM-MAC also has a *disadvantage* linked to the reception of SYNC messages from the border node. If the mobile node does not receive the SYNC message from the border node (due to losses on the medium), it may find itself in a virtual cluster with an unknown schedule. This leads to disconnection of the mobile node until it receives SYNC packets from a new border node and can follow a new schedule.

Authors in [PC15] propose a mobility support mechanism based on the exchange of SYNC messages sent by S-MAC - MS-SMAC. This solution allows mobile nodes to set adaptive connections with neighbors and keep low energy consumption. The authors add, besides the original SYNC message, still used between static nodes, a MSYNC message sent only by mobile nodes and an MACK message that is broadcasted by nodes that receive the MSYNC message. MSYNC contains information about the speed and the angle of the mobile node and are sent in broadcast, allowing other nodes to determine the presence of a mobile node in their neighborhood. MACK messages include the schedules that the static node follows (limited to maximum 2, in case of a border node).

In their solutions, authors consider perfect transmission of messages inside the communication range (which is fixed) and omit the impact of collisions. Considering this, the mobility of a node is determined here with the same tools as in MS-MAC, using received RSSI levels. When MSYNC messages arrive at the border router, it will reply with a MACK messages, letting the mobile node know also the schedule from the neighboring virtual cluster. Thus, as the mobile node moves into the new virtual cluster, it will know when to send its MSYNC messages.

MS-SMAC has an *advantage* that it gives mobile nodes a tool to notify neighboring nodes of their presence with MSYNC messages. Static nodes reply with MACK, thus the mobile node can adapt easily its schedule to the one in the neighborhood. This proposal also has a *disadvantage* when it comes to implement the solution in a real environment as it relies on the use the RSSI to monitor the mobility of nodes.

2.3.3 Frame-slotted protocols

Frame-slotted protocols manage mobility when mobile nodes enter the network by adapting the contention access or scheduled access periods in order to accommodate the mobile node (e.g. varying the scheduled access period time to accommodate transmissions from the mobile, keeping available slots in the scheduled access period for the mobile node).

The mobility-adaptive, collision free MAC (MMAC) [Ali+05], follows TRAMA [Raj+06] design principle. In a nutshell, TRAMA switches between random access and scheduled access (based on TDMA) periods. After the random access period, TRAMA pairs nodes that should transmit and receive during each slot in the scheduled access period. The access periods have a fixed duration that cannot be changed. MMAC adapts a flexible frame time, flexible transmission slots and flexible random-access slots. This will allow mobile nodes to enter the schedule following a predicted mobility estimation model. Averaging the location estimations is the basis of the node location prediction for the next frame. The mobile node transmits this estimation to the cluster head. Once the cluster head receives the estimation (it is always awake), it will broadcast it to other nodes in the last slot of the frame. All nodes in the cluster, after receiving the last slot in the frame, will know the position of the mobile node in the next frame. Now, static nodes can calculate independently how many neighbors they will have in the next frame and assess the frame duration. The cluster head also receives these estimations and computes a mean frame duration, which it will broadcast to all nodes. Nodes, once they receive the new frame time value and compare it with the one stored locally (the previous frame time) check to see if the new frame time is lower than the previous frame time. If this is the case, the random access interval is increased while the schedules access interval is decreased in order to keep the same frame time as before. The frame time can only be modified at the end of the round by using Global Synchronization Period (GSP). GSP averages the predicted frame duration from all cluster heads and distributes the new frame duration in the network to be used in the next round.

As *advantage*, MMAC can modify the size of the frame to accommodate a mobile node. However, this protocol has *disadvantages* as it is computational intensive and relies on historical data for the mobility estimation, making it unable to track rapid node movement inside current time frame. In addition, the adopted mobility pattern may not be valid in real scenarios, limiting the applicability of the protocol.

Another protocol in this class is the mobility aware TDMA-based MAC protocol for mobile sensor networks (M_TDMA) [JK07]. M_TDMA extends TDMA mechanism to mitigate changes in the network due to mobility. M_TDMA drops the global synchronization based on TDMA and partitions the network in non-overlapping clusters with one head per cluster. Nodes in clusters have a unique slot for transmissions. A round in M_TDMA is split into two parts, a control part and a data part. In the control part the adaptation for mobility is done, whereas in the data part, nodes exchange data packets. Not all slots are used for data transmission in the data part of the round, as a part of the last slots are left free for nodes that may enter further in the network.

In the first three slots of the round, which make the control part, information about cluster, node and slot assignment, are transmitted by the cluster head, one per slot. A node can determine if it is in the same cluster or in a new cluster by comparing information from the received cluster information slot with the stored information. If the node joins a new cluster, it will inform the head in the second round. The cluster head will check the available free slots at the end of the data part and, if more than one is available, it will assign a slot to the node entering the cluster. If only one slot is available, the head will halve the bandwidth for the new node, leaving space for any new node that may arrive later in the cluster (i.e. the cluster head assigns a half slot to the node). When no slots are available, the head halves further the available bandwidth. Once in the data part of the round, nodes exchange data packets using TDMA mechanism.

We can see that M_TDMA has as *advantages* a continuous integration of mobile nodes in the network by dynamically assigning slots in the frame. It maintains the same frame size and does not need any localization technique. There are also *disadvantages*, as mobile nodes are assumed not to leave the cluster during a round (information about the speed or the direction

of the mobile node is not available). Mobile nodes that do not receive the control part of the round must wait until they manage to receive a control part from a round, adding latency to communication.

M-LMAC [OMS09] is a protocol that supports mobility of nodes and is based on TDMA operations. Slots in M-LMAC are assigned to nodes by using a distributed algorithm described in [Nie+03] and not by a central manager, as it happens in the majority of TDMA based protocols. With M-LMAC, the mobile node, as it moves into a crowded area of the network, may experience collisions with other static nodes in its selected slot (as transmission slots of the mobile node may overlap after movement with the slots of a static node). A collision is detected as nodes, before sending the data packet, broadcast a message header in the control section, announcing the destination and length of the data packet. Collisions may occur as other static nodes can use the same frame as the mobile node, without knowing it. When the mobile node detects a collision, it will not transmit any more on that slot and will go in a *waiting for re-selection* mode. In waiting for re-selection mode, the mobile node can backoff several times, before finding a free slot to transmit its packets. Data packets will be sent without any acknowledgement. Authors propose to reduce the duration of the TDMA frame to 0.076 sec, as slots are long enough to send a packet of 250 bytes.

M-LMAC has as *advantages* that is does not need a central manager for frame allocation and that nodes can, in a distributed manner, determine when a mobile node is no longer in their neighborhood. This allows them to free the slot previously used by the mobile node. *Disadvantages* of M-LMAC come from the long disconnection of a mobile node if conditions in the network do not allow it to find a free slot to transmit its data packets. In addition, M-LMAC does not have any acknowledgement mechanism build-in when it makes data transmission, leaving reliability to upper layers.

2.3.4 Multichannel protocols

Until now, solutions that offer pure multi-channel mobility support are not present in the literature. Some attempts, such as MobiSense [Gon+11], use multi-channel support just to simplify network management and increase throughput, as cluster of nodes operate on different channels to reduce interference. Otherwise, MobiSense is a frame-slotted protocol that allows mobile nodes to join a static network similarly with other frame-slotted solutions.

Comparison of MAC protocols with mobility support in WSN In Table 2.2 we have an overview of existing solutions for mobility support in WSN. All available solutions focus on enabling communication from the mobile node to a static node. There is no solution envisaged that can provide mobility support for transmissions initiated by a static node and destined to a mobile node.

The asynchronous protocol class needs preamble messages to be sent before the actual data transmission, as they synchronize the sender and the receiver. This is a key advantage when mobile nodes are in the network. As there are no constraints regarding keeping synchronization, like they are present in synchronous or frame-slotted protocols, asynchronously protocols, are the prime candidates in providing seamless mobility integration in a network. Mobile nodes can overhear communication of other static nodes and use this to their advantage (e.g. wait for the transmission of static nodes and immediately send their data, steal communication channel or use a static node as an opportunistic forwarder). These operations lead to energy saving and can be easily integrated in network operations. The seamless integration of a mobile node is more important than providing equal transmission opportunities for all nodes, thus fairness is not always respected in asynchronous protocols. These characteristics make us choose the asynchronous protocols as the base for further development of mobility management solutions in this thesis.

Mobility support with synchronous solutions is largely based on S-MAC. The solutions try to include a mobile node in the network by updating the schedule of the mobile node. The neighbor discovery period in synchronous protocol is adjusted to track the mobility of nodes, thus static

Protocol	Advantage(s)	Disadvantage(s)
MA-MAC	Mobility based on threshold evaluation.	RSSI threshold can trigger more handovers than needed.
MoX-MAC	Mobile node opportunistically sends data packets to static nodes.	Mobile node has no guarantees that it can reach the sender of the preamble.
X-Machiavel	Mobile node can always send data packets. High energy efficiency.	Unfair for static nodes, that experience delays in transmission
MS-MAC	Continuous connection for mobile node during search for new attachment point.	All nodes participate in mobility management. Neighbor discovery reduces available energy for all nodes. Relies on RSSI threshold to determine a node mobility.
MD-SMAC	Only mobile node pariticpates in mobility management. Mobile node can lower neighbor discovery periodicity after moving to a new cluster.	Relies on RSSI threshold to determine a node mobility.
AM-MAC	Mobility determined by received SYNC message from border node.	Mobile node becomes disconnected if SYNC messages from the border node are lost.
MS-SMAC	Mobile node informs neighbors of its presence with MSYNC messages.	Relies on RSSI threshold to determine a node mobility. Idealized network conditions.
MMAC	Flexible frame time includes mobile node transmissions.	Computational intensive. Movement prediction based on historical data. Invalid mobility pattern in real scenarios.
M_TDMA	Dynamic slot assignment for mobile nodes. No localization technique needed.	Bandwidth reduction once new mobile nodes enter the neighborhood. Mobile node disconnected if no control part is received.
M-LMAC	No central manager needed to allocate transmission slots.	Long disconnections for mobile node when no free slot is available. No build-in acknowledgement mechanism.

Table 2.2: MAC protocols with mobility support

nodes participate in the mobility management process. The static nodes can be relieved of this task if the border node provides itself mobility management information (schedules of virtual clusters) for the mobile node. Keeping synchronization between mobile nodes and neighbors is challenging and it increases energy consumption for nodes. Missing synchronization packets (e.g. due to collisions), especially when the mobile node changes clusters, can disconnect the mobile node from the network. Some of the presented mechanisms rely also on received RSSI values, which are not a reliable trigger for mobility detection. In our opinion, the synchronous protocols class, given these characteristics, does not provide efficient mobility management in WSN and will not be further used or evaluated in this thesis.

The next class of protocols where mobility solutions have been developed is the frame-slotted class. In frame-slotted solutions for mobility management, a mobile node is included in the transmission schedule either by changing the duration of scheduled access period (e.g. in MMAC) or by keeping slots free in the scheduled access period (e.g. in M_TDMA). Mobility detection does not always keep up with the movement of mobile nodes, as it relies on historical data of the position of the mobile node. The delay in communication between frames (as the mobile node will need to wait for its slot to send the data packet) can cause collisions when the mobile node moves in an area with static nodes that use the same frame slot, but the schedule is not updated. Performances can also be degraded as the mobile node can receive transmission slots with lower transmission capabilities (e.g. lower bandwidth). Again, given the drawbacks enumerated before, we believe that frame-slotted protocols cannot provide an efficient support for mobility management and will not be used or evaluated in this thesis.

The multichannel class of protocols does not have a dedicated solution, the only available protocol, MobiSense, using multichannel support just to simplify network operations. MobiSense relies on frame-slotted operations similar to ones that we have discussed before for transmission related operations. This class of protocols will again not be used in this thesis, as it does not have any dedicated solution for mobility management.

2.4 Conclusion

This chapter makes an overview of existing solutions for mobility at MAC layer. Proposed solutions from each category present good performances for pre-defined topologies and traffic patterns. Adaptations that allow mobile nodes to communicate with other nodes in the network have also been made. We have seen complex solutions for mobility management (such as MMAC), which are too computational intensive to be deployed in a real environment. Out of all the asynchronous protocols, we have chosen X-Machiavel, as it provides easy integration of mobile nodes in WSN. It ensures transmission priority for mobile nodes regardless of conditions in the network. Energy efficiency for both the mobile and static nodes is comparable to X-MAC, and computations are kept at the minimum. There is no need for synchronization besides exchanging preambles. The mobility of the node is inferred from the received preamble. Transmissions from mobile nodes have priority in the network, as packets from the mobile node are sent with a different preamble. X-Machiavel provides thus the most complete solution, having an end-to-end approach, broader than the one hop approach of the other solutions. That is why X-Machiavel is the MAC protocol of choice for mobility management in this thesis. We will leverage later (Chapter 5), X-Machiavel's capabilities to achieve efficient mobility management at the routing layer.

Still, it is our belief, shared with authors of [Kun10], that mobility management should be a joint effort at both MAC and networking layers (namely routing protocols), especially if we consider communication from a static node to a mobile node. The main problem is not the communication from the mobile node to the static node, for which we have seen many solutions, but the communication from the static node to a mobile node, where a routing protocol is needed. We will focus in Chapter 5 on integrating X-Machiavel with a dynamic routing protocol, as opposed to the static routing used with X-Machiavel in [Kun10], making thus a step further towards seamless integration of a mobile node in the network.

In the next chapter we will explain how routing protocols work, we will present existing mobility solutions at the networking layer as well as an overview of a dynamic routing protocol specifically designed for WSN that is, in our opinion, the best candidate for mobility management at the networking layer.

Chapter 3

Network layer mobility support in WSN

3.1 Introduction to the WSN network layer

We have seen in the previous chapter that solutions to communicate between nodes exist at the MAC layer. They ensure communication with nodes in one's neighborhood. Nodes usually relay their information towards a central point in the network, the sink. One node may increase its transmission power to communicate directly with the sink, or it can involve other nodes in a collaborative way to ensure communication with the sink. The first option is not a good fit in WSN, as energy efficiency is a core requirement in such networks and increasing the transmission power will consume more energy. The second option requires that intermediate nodes, between the source node and the sink, propagate the packet over multiple nodes, or hops, until the sink is reached. This is the way communication is established in a WSN, using intermediate nodes to relay information between a source and a destination (i.e. this is called hop-by-hop transmission).

At first, simple solutions have emerged that allowed nodes to relay information between themselves and a sink: flooding and gossiping. Using the flooding algorithm, the sender node broadcasts packets in the neighborhood and any receiver retransmits the packets in the same way, until eventually, the packets arrive at the destination. We can see that flooding is a simple idea, but has a major drawback as implosion occurs (nodes redundantly receive multiple copies of the same message [GV+09]). Nodes will resend copies of the packet in the network, leading to packet duplication at other nodes. This technique impacts greatly the energy consumption of nodes. With flooding, nodes send packets blindly, not knowing who will receive them. An improvement of the flooding algorithm is the gossiping algorithm [Zan+07]. Instead of broadcasting the packet, a node will randomly choose one neighbor to which it will send the packet. This avoids the implosion problem found in flooding algorithm. However, some of the problems found in the flooding algorithm remain, as nodes do not keep track of whom they sent packets, so they are still send blindly. The same node may receive several times the same packet if it happens to be chosen by other neighbors.

These proposals are not efficient in sending data packets in an energy constraint environment as they waste energy. Considering this, routing protocols become necessary in Wireless Sensor Network. Still, running a routing protocol in WSN poses challenges [GV+09]. To build routes, routing protocols need unique identifiers of nodes. Unique identifiers for WSN nodes were expected to be developed, as a framework for new network abstractions has been provided [Lev+04]. For example, in the beginning Active Message Dispatch ID were used as unique identifiers in TinyOS [Lev+05] instead of IP addresses.

Since those beginnings, the research community has gained more knowledge on how WSN networks were used in real implementations. A move toward IPv6 addresses was envisaged, as the huge address space (2^{128} available addresses) provides the necessary unique addresses for current and future devices, even as the number of connected objects is expected to reach billions

of devices in our lifetime [Bus14]. Authors of [Dun03] propose uIP, proving that IPv6 addressing was feasible in WSN. As IPv6 packet size is unsuitable for WSN characteristics (if we consider the 127 bytes constraints of IEEE 802.15.4 [802a]), an adaptation layer, 6LoWPAN, has been defined to carry the meaning of IPv6 addresses in a compact form using IEEE 802.15.4 short address [Mon+07].

6LoWPAN, in a nutshell, is an adaptation layer that enables transmission of IPv6 packets in a WSN (especially using IEEE 802.15.4). To achieve this, 6LoWPAN makes a stateless header compression of IPv6 and UDP headers and makes fragmentation and reassembly of IPv6 packets. 6LoWPAN has low processing and storage costs. Header compression is achieved by eliding unnecessary fields that have known values (e.g. version number in IPv6 address) or that can be inferred from other information (e.g. source and destination addresses in IPv6 header that can be derived from the link address). Other fields in the header are compressed using tables of correspondences (e.g. next header in IPv6 header). 6LoWPAN manages to compress a 40 bytes IPv6 header in only 3 bytes. As packets on IEEE 802.15.4 links cannot exceed 127 bytes and 25 bytes are already occupied for IEEE 802.15.4 headers [802a], this leaves little space for data. Large packets from the upper layers (above 6LoWPAN) need to be fragmented if they exceed the maximum available data size in an IEEE 802.15.4 packet. 6LoWPAN fragments large packets by putting a tag in the fragments (the same tag in all fragments of the same packet) to ease reassembly. The receiver waits all fragments before attempting reassembly of the IPv6 packet.

Some reasons why IPv6 was considered over IPv4 can be found in [HC08]. Authors militate for IPv6 as it has generality and extensibility that allows using mechanisms already available for WSNs. Trickle-based dissemination, sample-listening, hop-by-hop feedback or collection routing are some of the mechanisms that ease IPv6-based network architecture implementation. IPv6 address structure also allows cross-layer compression, achieved with 6LoWPAN. Address auto-configuration, and Internet Control Message Protocol version 6 (ICMPv6) messages are important features of IPv6, enabling scalability and unattended operations. These features are also equally important in any WSN. Here is why, IPv6 is used in WSN networks, also providing unique identifiers for routing protocols.

Up until recently, communication in WSN has been done mainly between static nodes. The network, once deployed, builds logical topologies that remained more or less stable during the lifetime of the network. Transient links, or the disappearance of a node can be mitigated by the majority of protocols, both at MAC and routing layer. Although static networks can cover a broad spectrum of applications ([Bha+07], [WA+06]), some applications, such as target tracking or wildlife monitoring ([All+07], [Leg+08], [Sik+06]), require mobile nodes. We can distinguish two types of mobility that are present also at the network layer: *roaming* and *handover* [KP06]. Services that want to be offered in a different place than where they were registered, as devices are kept connected to the network, are in a roaming process. On the other hand, when a handover is done, devices change their point of attachment in the same network, without communication interruptions.

In this chapter, we focus on handovers done at the network layer, as physical movement of nodes, corroborated with their limited transmission capabilities, will determine a change of attachment point in the network as they move. First, we start making a classification of routing protocols. After this, we will focus our attention on solutions that already provide mobility support in standard IP networks, such as MIPv6 and its extensions. These standards cannot be used directly in WSN, as adaptations are need to meet the specific requirements of WSN. Before moving forward with this approach, we conduct an analysis of IPv6 address auto-configuration in WSN from a theoretical point of view. Address auto-configuration is a critical component in MIPv6 solutions, as it allows mobile nodes to obtain a valid IPv6 address in any new network where they might move. Only after a valid IPv6 address is obtained, MIPv6 actions can be performed. Our findings point out to potential problems, such as the large size of messages sent during IPv6 address auto-configuration, that will not fit in IEEE 802.15.4 frame even using 6LoWPAN compression. This will render unusable such solutions in WSN. Thus, we turn our attention to another standardized solution, IPv6 Routing Protocol for Low-power and Lossy Networks (RPL). After explaining how RPL works, we close this chapter with an overview of the

state of the art on RPL-based solutions to support mobility.

3.2 Classification of routing protocols

In Wireless Sensor Network, data is transmitted on a hop-by-hop basis, so several intermediary nodes may compose the route towards a destination. Nodes have an IPv6 address, threfore when a routing protocol is used, the node should also know where (or to whom) to send its data packets. That is why routing in WSNs has been extensively studied in the past [AY05]. In small networks, flat routing schemes are sufficient, but as the network size grows, geographical or hierarchical routing protocols take over the routing task. Hierarchical protocols can be further divided into proactive (table-driven), and reactive (on demand) protocols.

Flat routing protocols typically use data centric routing (i.e. do not use unique identifiers) and all nodes have the same role [AKK04]. For the purpose of this thesis, flat routing protocols do not offer the necessary support for the envisaged mobility of nodes, where a change of attachment point is needed. That is why we will not include flat routing in any further analysis.

Geographical routing protocols formulate an efficient route towards the destination by using the location information available at the nodes.

Reactive protocols aim is reducing the control packets when a very low traffic has to be routed in the network.

Proactive protocols aim is constructing a priori efficient routes.

The last three types of routing protocols have the possibility to integrate mobile nodes in the network and will be closer analyzed in the following.

Geographical routing protocols Using location information, geographical routing protocols [Kar+12] are suitable for large multi-hop sensor networks where topology changes occur frequently and nodes are not reliable. Routing is done on a hop-by-hop basis, using only one-hop neighboring nodes. One of the best-known protocols is Greedy Perimeter Stateless Routing (GPSR) [KK00]. Nodes use a simple beaconing algorithm and broadcast periodically their position and identifier. Greedy and perimeter forwarding are the two forwarding methods used. The first method allows packets to be forwarded to the neighbor closest to the destination (as seen in Figure 3.1 where node X sends data to node Y, as node Y is the closest to destination node D), but if no neighbor is available, or the packet has not reached the destination, the second method of forwarding is used. Perimeter forwarding allows the packet to traverse a possible void between the source and destination.

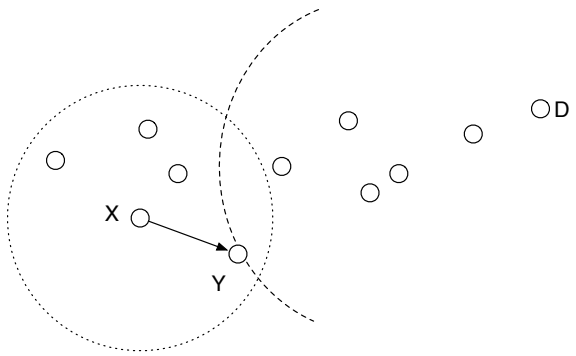


Figure 3.1: GPSR greedy forwarding

Geographic routing protocols have *advantages* when it comes to high scalability and mobility support, as routing is done on a hop-by-hop basis. Overhead for communication remains minimal even in high-density networks [Kar+12]. *Disadvantages* are linked to routing loops

that can form in the network, as only one-hop information is available at nodes. In addition, the acquiring of location information may involve high-energy cost (e.g. when using Global Positioning System (GPS) devices).

Reactive routing protocols Reactive routing protocols for WSN have taken inspiration from MANET reactive protocols such as Ad hoc On-Demand Distance Vector (AODV). AODV reduces routing load, uses table driven routing mechanism, has location independency and uses destination sequence numbers for routing packets to destination [Kas+11]. Direct operation of AODV in WSN is inefficient, as it needs periodic beaconing for information like source and destination sequence number, so protocols like 6LoWPAN Ad Hoc On-Demand Distance Vector (LOAD) [Kim+07] lighten AODV operations. LOAD enables multi-hop routing between IEEE 802.15.4 nodes. It operates on top of the adaptation layer (6LoWPAN) and should be only run on Full Function Devices (FFDs). Routes between nodes are built in LOAD using the Link Quality Indicator (LQI) metric obtained from the physical layer or using the hop count. There are several divergent paths from AODV. LOAD drops the destination sequence number from AODV, as intermediate devices cannot reply with a *route reply* (RREP in Figure 3.2) to a *route request* (RREQ in Figure 3.2), even if they have an active route for the intended destination. Only the destination can reply to route requests (RREQ) with route reply (RREP).

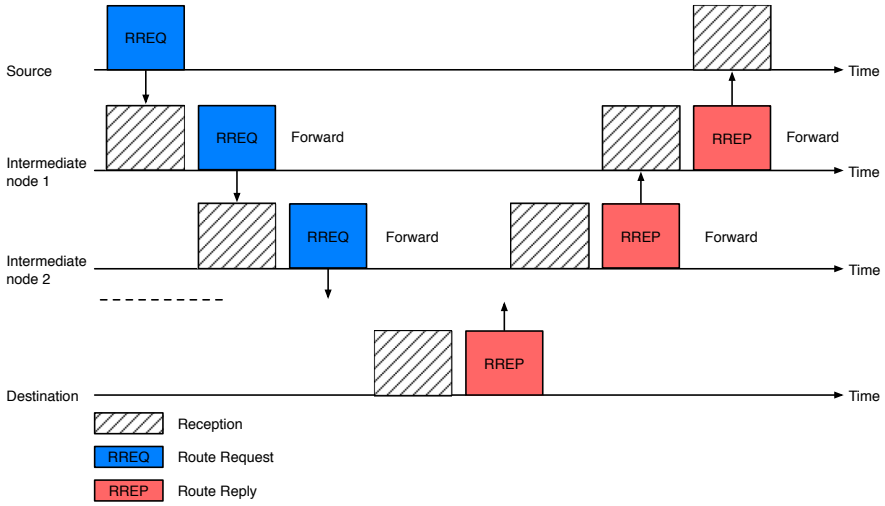


Figure 3.2: LOAD route discovery

Reactive routing protocols have the *advantage* that routes are build only when needed. Control traffic is thus present just when routes need to be discovered, reducing overhead for communication. However, some *disadvantages* come from the added latency during initial route construction and that packet flooding is done during route discovery.

Proactive routing protocols Preparing routes before they are needed, proactive routing protocols maintain routing information between neighbors by sending constantly control packets in the network. Proactive routing protocols compute routes in general using parent-child relationships, based on a graph. One of the first attempts to build a proactive routing protocol for WSN is the Collection Tree Protocol (CTP) [Gna+09], a distance vector routing protocol. It computes anycast routes to a sink, or the root of the routing tree. CTP strives to achieve datapath validation (i.e. using data packets to dynamically probe and validate the consistency of the routing topology) and adaptive beaconing (i.e. extension of trickle algorithm so that in can be applied to routing control traffic) by meeting four goals: *reliability* - deliver at least 90% of end-to-end packets when a route exists, mainly without end-to-end mechanisms, *robustness* - configuration free deployment on networks

with different topology, different load or in different environments, *efficiency* - limited state maintained to deliver packets and *hardware independence* - meet all before mentioned demands without needing specific features on sensor nodes [Gna+09]. Datapath validation embeds in data packets the needed information to maintain routing topology and detect routing loops. Adaptive beaconing extends the trickle algorithm [Lev+11], presented later in this chapter, to dynamically adapt control traffic.

Inspired from CTP, the IETF ROLL working group standardized the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [Win+12]. It is a scalable approach that offers IPv6 connectivity to sensors networks which can contain several hundreds interconnected devices. In contrast to CTP, RPL supports also downward routes - point-to-multipoint traffic (from the sink towards the nodes), and point-to-point (communication among the nodes). Routes in RPL can be computed with a variety of metrics, ranging from hop count, to LQI or Expected Transmission Count (ETX) [DC+03].

Proactive routing protocols present *advantages*, as routes are constructed before they are needed, and are continuously updated. The routes are thus always available to nodes. In addition, latency during route setup is lower when compared to reactive protocols. *Disadvantages* come from the constant overhead in communication, as control packets must be periodically sent in the network to maintain the routing topology.

Routing Protocol type	Advantage(s)	Disadvantage(s)
Geographical	Provides easy integration of a node in the network, as the location of neighbors is know in advance (through periodic beaconing).	Needs additional equipment (e.g. GPS device) to provide accurate location information. Loops can form in the network, as only one hop information is considered in routing decisions.
Reactive	Construct routes only when needed, so control traffic is sent just during this process.	Introduces latency during initial route construction. Packet flooding during route construction.
Proactive	Maintain route information between neighbors, regardless of traffic. Routes are always available before traffic is sent.	Control traffic is always present in the network in order to maintain up-to-date route information.

Table 3.1: Routing protocols in WSN

Comparison of routing protocols in WSN This short taxonomy of routing protocols, summarized in Table 3.1, makes easier the choice of a routing protocol to support mobility. Geographical routing protocols can integrate a mobile node in the network, but there is a need to know the location of nodes, either using an energy consuming device such as a GPS or through complex calculations that are computational intensive. Computed routes are not always efficient, as loops can form due to limited information available for routing (only one-hop information is available at the node). Applying geographical routing protocols in WSN networks has to be done with care, as energy consumption is a major concern. Mobile nodes will deplete their energy resources faster than other nodes, as they will need to update their position more frequently than static nodes, regardless of the used technique. Reactive routing protocols build routes only on demand, and limit the use of control traffic. Nevertheless, a mobile node in the network, as it changes frequently its attachment point would require constant tracking, inducing regular control traffic. Considering that route construction is done mainly through packet flooding in reactive routing protocols, each time a mobile node requires the construction of a new route, nodes in the neighborhood will pay an energy penalty, reducing network lifetime (i.e. the period of time

a network is running before the first device depletes its energy resources and disconnects from the network). Proactive routing protocols, keep updated all routes in the network by sending constant control traffic. Solutions specifically designed for WSN have mechanisms that limit the control traffic, such as the trickle algorithm, allowing energy savings for nodes. Having a global overview of the network, proactive routing protocols can eliminate easier routing loops than geographical routing protocols and always try to compute efficient routes (using metrics such as ETX, LQI and others, as dictated by the envisaged route optimization). A mobile node in the network can thus take advantage of up-to-date routes kept by proactive routing protocols and find easier new attachment points in the network. Mobility using IP addresses can also take advantage of routing protocols.

In the next section, we take a look over MIPv6 and its extensions. MIPv6 is the standard that provides mobility support in IP networks. The protocol cannot be used directly in WSN, as control messages that are too big to be transmitted in WSN are present in MIPv6. Nevertheless, there are optimizations for WSN that might make feasible implementing mobility support solutions such as MIPv6 in WSN. These optimizations are linked to a new version of Neighbor Discovery that eliminates the need of duplicate address detection or to 6LoWPAN frame header compression.

3.3 Mobility management at IP level

In an IP network, as nodes move, they can change their point of attachment to the network. This change can be done seamlessly using standard protocols designed to handle mobility such as MIPv6. Early versions of this protocol date back to 2004, and are one of the first attempts to provide mobility management for IPv6. There are other variations of mobility management, like Proxy Mobile IPv6 (PMIPv6) [Gun+08] or Hierarchical Mobile IPv6 (HMIPv6) [Sol+08], but they have at their core operations similar to the ones present in MIPv6. This is why we chose to present only how MIPv6 manages the mobility of nodes, as it is the most prominent exponent of them all.

MIPv6 allows nodes to remain reachable, regardless of their position in IPv6 networks. Without any support for mobility, nodes that move in a new location will become unreachable. In the new location, the mobile node could obtain a new IPv6 address, but then, the mobile node would not be able to maintain transport and higher-layer connections when it changes location. MIPv6 enables a mobile node to move between links without changing the mobile node *home address* (i.e. the IPv6 address obtained after the node is powered on the first time). Packets may be routed to this address of the mobile node, regardless of the current mobile node's attachment point to the Internet. The mobile node will also be able to communicate while away from its home link, making mobility transparent for higher-layer protocols [Per+11].

To achieve this, MIPv6 uses a relay station for packets sent or destined to a mobile node, called the *Home Agent (HA)*. After the node is started, it will consider the network where it resides as the *home network* and will acquire an IPv6 address from a router. The HA is an entity that will manage connections to and from the mobile node while the mobile node is away from the home network. It will register and update *bindings* for the mobile node, so that it will be able to redirect any traffic that is destined to the mobile node, while the mobile node is away from its home network. Bindings are associations of IPv6 addresses, where one address is the home address of the mobile node and the second one is the IPv6 address in the mobile node's new location (called *care of address*). As the mobile node has movement capabilities, it can find itself in a new network, called *visited network*, different from its home network. The mobile node will detect its movement away from the home network when it receives a router advertisement from routers in the visited network. Once the router advertisement is received, the mobile node, through IPv6 address auto-configuration, will acquire a valid IPv6 address in the visited network - the *care of address*, from the router in the new network. Once the node moves into a visited network, it needs to notify his HA of the new attachment point. The mobile node will send a *binding update* message to the HA, informing it of the new care of address in the visited network.

The HA will create or update an entry in the binding cache. Registering on a visited network is illustrated in Figure 3.3.

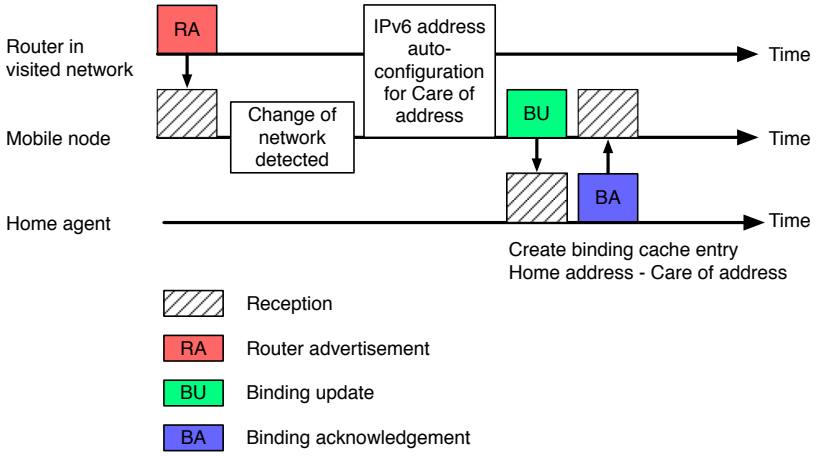


Figure 3.3: MIPv6 care of address registration

Using the Home Agent, the mobile node will remain reachable when they are in visited networks. Communicating with outside nodes, named *correspondent nodes*, can be done in several ways:

- If the mobile node is in its home network, communication with a correspondent node works as in a regular IP network. The home agent will not have any binding cache entry for the mobile node and will not be involved in communication.

- Once the node enters a visited network, packets from the correspondent node are addressed to the home network of the mobile node. Packets arriving in the home network, and destined to the mobile node are routed to the home agent and then tunneled to the mobile node. The home agent uses proxy Neighbor Discovery to intercept any IPv6 packets sent to the mobile node's home address on the home link. Each intercepted packet is tunneled to the mobile node care of address using IPv6 encapsulation [CD98].

- The mobile node can register the care of address at the correspondent node (in the binding cache of the correspondent node, if the correspondent node has one) and so, in a *route optimized* way, packets will travel directly between the two entities (with the help of a new type of IPv6 routing header, proposed in [Per+11]).

Various attempts to port existing protocols for mobility in IP networks to WSN have been made. Authors of [Rot+12] provide experimental results for MIPv6 [Per+11] implementation over a 6LoWPAN network and conclude that MIPv6 could be implemented in a WSN, as MIPv6 related operations take little time, with movement detection being one of the greatest problems to address.

Our belief is that this problem could be addressed by the latest optimizations of Neighbor Discovery [She+12]. Our first contribution in the thesis is an analysis of IPv6 address auto-configuration, using the latest optimizations available for WSN from a theoretical point of view. Getting a new IPv6 address in an efficient way in a new location should speed up the movement detection processes in MIPv6.

3.4 IPv6 address auto-configuration

3.4.1 Neighbor Discovery in the IPv6 protocol suite

Before exchanging information on an IP network, nodes should gather configuration information (such as IP address, network prefix, default router, etc.). IPv6 anticipated the need to configure and manage a large number of nodes by supporting stateless address auto-configuration. This mechanism requires no manual configuration of hosts, minimal (if any) configuration of routers and no additional servers, which make it particularly practical in the context of WSN.

Neighbor Discovery [Nar+07] is one of the major protocols included in the Internet Protocol Suite. Similarly to the Address Resolution Protocol (ARP) in IPv4 networks, Neighbor Discovery allows the discovery of other IPv6 hosts on the link. Neighbor Discovery also includes many improvements over ARP such as router discovery, packet redirection, maintenance of reachability information about active IPv6 neighbors, address auto-configuration and duplicate address detection. For this, Neighbor Discovery defines new ICMPv6 packets known as Router Solicitation (RS) and Router Advertisement (RA), Neighbor Solicitation (NS) and Neighbor Advertisement (NA), and Redirect. Redirect messages are used by router to inform hosts of better route for packets sent to a particular destination. The purpose of Neighbor Solicitation and Neighbor Advertisement are twofold. First, Neighbor Solicitations are used to resolve the link-layer address of a neighbor (similarly to ARP in IPv4). Each node registers the discovered correspondence between IPv6 and link-layer addresses in its neighbor cache. Neighbor Solicitations also serve to verify that a known neighbor is still reachable. Neighbor Advertisements are sent in response to Neighbor Solicitation message.

Routers periodically send Router Advertisements to advertise their presence on the link and spread various parameters about the network connectivity. Hosts not willing to wait for the next scheduled Router Advertisement can send Router Solicitation to trigger the transmission of a new Router Advertisement. The reception of a Router Solicitation allows a host to discover the address of the router, the IPv6 prefix(es) of the link together with the associated lifetime(s), and in some case the address of a Domain Name Server (DNS) server. With such information, a host is able to configure its own global IPv6 address by prepending the received IPv6 prefix to the 64 bits Extended Unique Identifier (EUI-64) of its network interface. This new address is associated to the interface as tentative. Before final association, the host has to verify its uniqueness on the link by performing duplicate address detection. This phase consists in trying to discover hosts that already own this IPv6 address by sending Neighbor Solicitations. A host already using this address replies with a Neighbor Advertisement. In that case, the address cannot be assigned to the interface of the original host. If there is no response after the transmission of three consecutive Neighbor Solicitations, the host assumes that the address is unique and can be assigned to its interface.

When an IPv6 address is needed for a new host, there is already a proposed protocol, Neighbor Discovery, that provides stateless auto-configuration with minimal or no interactions. Nevertheless, in WSNs one of the main characteristics is the multi-hop nature of communication. This in turn requires some fine-tuning to the IPv6 address auto-configuration protocol defined for IPv6 in part due to the lack of a common link between nodes. In addition, WSNs may not strictly follow the defined concept of subnet and links for IP networks. We can think of a situation when two neighbors in WSN belong to different IPv6 subnets or of links that have asymmetrical reachability. Neighbor Discovery uses heavily multicast signaling for control packets, but in WSN, we try to limit multicast signaling use due to energy conservation considerations. Neighbor Discovery, as defined for IPv6 networks, is not suited for WSN. This is why the IETF has defined an optimized version of Neighbor Discovery [She+12], which enables stateless address auto-configuration. All adaptations of IPv6 for WSN proposed so far do not consider the optimized Neighbor Discovery version.

3.4.2 Optimizations of Neighbor Discovery for WSN

The optimizations to Neighbor Discovery for WSN [She+12] introduce two actors known as the Border Router (6LBR) and the Router (6LR). The 6LBR connects the network to another IPv6 network and is responsible for spreading the IPv6 prefix(es) inside the WSN. The 6LR is an intermediate node that extends the range of the network. Two topologies are envisioned for interconnecting hosts and routers - mesh under and route over, as seen in Figure 3.4. In mesh-under, the hosts are connected to a 6LBR through a mesh, using link-layer forwarding. Each host is therefore one IP hop away from the 6LBR regardless the number of layer 2 transmissions needed to reach the 6LBR. In other words, this topology simulates the traditional IP subnet topology with one router and multiple hosts in the same subnet. By contrast, route-over introduces a topology where hosts are directly connected to routers (being 6LR or 6LBR). Each link consists in an IP hop and packets are transmitted using IP routing. In the next sections, we focus on the route-over topology.

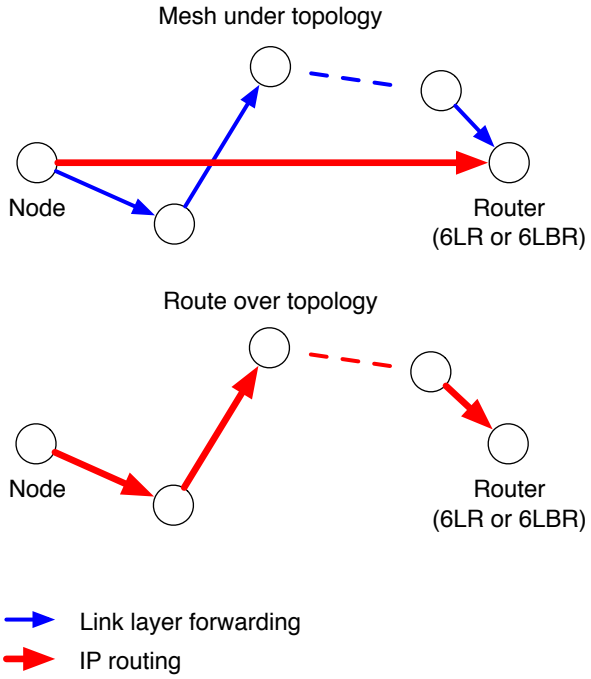


Figure 3.4: Mesh under vs. route over topology

Router discovery still uses router solicitations and advertisements but routers do not need to send periodic router advertisements since hosts will always solicit information by sending router solicitations. However, periodic router advertisements might still be used to distribute prefix information between routers over a route-over topology. Router solicitations are sent either in multicast to discover new routers (e.g. at bootstrap, hosts do not know yet any routers) or in unicast to solicit updated information from a known router. By contrast, router advertisements are now mainly sent in unicast to the originator of router solicitation.

The proposed optimizations also introduce a new address registration mechanism to remove the duplicate address detection and centralize the discovery of neighbors. Routers are provided with the address of hosts in order to determine whether an address is already used and to resolve link-layer addresses on behalf of hosts. When a host has configured a new global IPv6 address (usually upon reception of a new router advertisement), it registers this address with one or more of its default routers by transmitting a neighbor solicitation. Neighbor cache of routers is therefore extended with new status such as registered (entries that have a registered lifetime) and tentative (temporary entries with short lifetimes). Upon reception of a neighbor solicitation

demanding the registration of an address, the router first inspects its neighbor cache to see if the claimed address is unique. If this verification does not result in a duplicate address being detected, the router creates (or updates) an entry for the IPv6 source address of the neighbor solicitation in its neighbor cache. Then, a unicast neighbor advertisement is sent to the host with the status of the registration (success, duplicate address, etc.). An illustration of address registration is presented in figure 3.5.

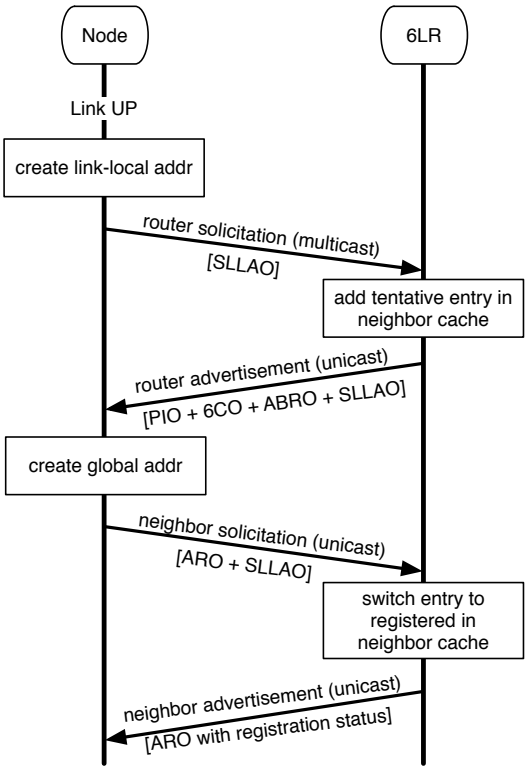


Figure 3.5: Address registration in optimized Neighbor Discovery

As part of the optimizations, address resolution is no longer performed by multicasting neighbor solicitations. It is assumed that link-local addresses are formed from the EUI-64. Packets destined to the link-local prefix are always sent on the link to that destination by reversing the procedure of creating the EUI-64. All other IPv6 prefixes are always assumed off-link. So packets destined to off-link destinations are sent to one of the default routers. Then the router uses its routing table to determine the next-hop IP address and so on. Finally, only hosts perform neighbor unreachability detection in order to verify that their default routers are still reachable. For this, hosts send a unicast neighbor solicitation and waits for the reception of the corresponding unicast neighbor advertisement. The optimizations presented above require new options known as Address Registration Option (ARO), 6LoWPAN Context Option (6CO) and Authoritative Border Router Option (ABRO). ARO are included in Neighbor Solicitations in order to register with routers IP addresses of hosts that are directly reachable. Their corresponding link-layer addresses are also included in such neighbor solicitation via a Source Link-Layer Address Option (SLLAO). Registration status is also transmitted to hosts via ARO as part of neighbor advertisements. 6CO carries prefix information for 6LoWPAN context-based header compression [HT11]. Furthermore, this option allows for the dissemination of multiple contexts identified by a 4 bits Context Identifier (CID). A context may be an IPv6 prefix or an IPv6 address. 6CO could be included in router advertisements in addition to the well-known Prefix Information Option (PIO). ABRO carries information from the authoritative 6LBR. ABRO is included in all router advertisements when those messages are used to disseminate prefixes and context information

across a route-over topology. Such option allows 6LR to maintain up-to-date information on prefix and context information.

In the next section we analyze the underlying layers, to see how they will impact Neighbor Discovery messages exchanged between nodes at networking layer.

3.4.3 In depth analysis of non-beacon IEEE 802.15.4

PHY and MAC layers

In the following, we consider that nodes use the IEEE 802.15.4 standard with the 2.4 Ghz band using the Offset-Quadrature Phase Shift Keying (O-QPSK) modulation. Physical payloads (packetSize) are prefixed with a synchronization header (SHR) and a physical header (PHR). The duration of a transmission ($phyFD$) is given by:

$$phyFD = \frac{SHR + PHR + packetSize}{D} \tag{3.1}$$

where SHR, PHR and packetSize are expressed in bits and D denotes the bit rate in bits/s regarding the physical layer in use (frequency band and modulation). IEEE 802.15.4 frames have a maximum size of 127 bytes (packetSize) in which 25 bytes are taken by the MAC header (when 64 bits extended address and no auxiliary security header are used). This leaves a maximum size of 102 bytes for the data. By contrast, acknowledgments are only 5 bytes long. Table 3.2 illustrates the parameter values corresponding to the 2.4 Ghz band.

If the sender requests the receiver to confirm the reception of the data, the sender should receive an acknowledgement after the successful transmission of the data. Before transmitting acknowledgment, the receiver should switch from RX to TX. The RX to TX turnaround time should be less or equal than $turnTime$. As a result, once a node has access to the medium, the duration of the transmission FD is expressed as follows:

$$FD = \frac{2(SHR + PHR) + packetSize + ackSize}{D} + turnTime \tag{3.2}$$

In non-beacon mode, IEEE 802.15.4 nodes relies on unslotted CSMA/CA to access the medium. Once a node wants to send a data frame, it first draws a random number $x \in [0, 2^{BE-1}]$ (Backoff exponent (BE) being the backoff exponent) and delays its transmission for x backoff periods. Then, the node checks if the medium is free by performing a Clear Channel Assessment (CCA). If this verification does not result in a busy medium, the node can send its frame. If the medium is busy, the node should schedule a retransmission by increasing BE. BE is stacked between a minimal and maximal values referred to as minBE and maxBE. A node can schedule up to maxCSMABackoffs trials before declaring a channel access failure. Therefore, the minimal duration of the backoff procedure (minBackoffDuration) is equal to 0 (i.e. $x = 0$ and the medium is free at the first attempt). The maximal duration of the backoff procedure (maxBackoffDuration) is illustrated in Equation 3 (where $m = min(maxBE - minBE, maxCSMABackoffs)$). In conclusion, the total minimum time δ_{min} (respectively maximum time δ_{max}) required to transmit a frame without errors is expressed as follows:

$$maxBackoffDuration = \left[\left(\sum_{k=0}^{m-1} 2^{minBE+k} \right) + (2^{maxBE} - 1) \times (maxCSMABackoffs - m) \right] \times BackoffPeriod \tag{3}$$

$$\delta_{min} = FD \tag{3.3}$$

$$\delta_{max} = maxBackoffDuration + FD \tag{3.4}$$

Wireless communications are particularly affected by radio interference, disconnection and contention level in radio cells. In that context, we need to take into account the probability of transmission errors. Let E_r denote the bit error rate and P_c denote the probability to successfully transmit a data and receiving the corresponding acknowledgement. P_c is therefore expressed as:

$$P_c = (1 - E_r)^{(2\times(PHR+SHR)+packetSize+ackSize)} \tag{3.5}$$

The probability of a successful transmission (both data and ack) at the i^{th} trial is as follows:

$$(1 - P_c)^{(i-1)} \times P_c \tag{3.6}$$

As a result, the minimum average time (δ_{avg_min}) required to successfully transmit a data is given by:

$$\delta_{avg\_min} = \sum_{i\geq 1} \delta_{min} \times i \times P_c \times (1 - P_c)^{(i-1)} = \frac{\delta_{min}}{P_c} \tag{3.7}$$

Similarly, the maximum average time required δ_{avg_max} required to successfully transmit a data is:

$$\delta_{avg\_max} = \frac{\delta_{max}}{P_c} \tag{3.8}$$

Attributes	Values
Synchronization header (SHR)	5 bytes
Physical header (PHR)	1 byte
ackSize	5 bytes
aUnitBackoffPeriod	32 μs
turnTime	512 μs
Symbol Duration	16 μs
minBE	3 (default)
maxBE	5 (default)
maxCSMABackoffs	4 (default)
bit rate (D)	250 kbits/s
symbol rate	62.5 ksymbols/s
MSP 430 processor	1.8mA (Active)
	5.1μA (Sleeping)
CC2420 transmission	8.5mA (at -25dBm)
CC2420 reception	19.7mA

Table 3.2: Physical and MAC attributes regarding IEEE 802.15.4

We have seen how the IEEE 802.15.4 MAC and PHY parameters influence communication between nodes. Next, we will analyze how Neighbor Discovery messages fit in IEEE 802.15.4 frames and what improvements can 6LoWPAN bring to communication.

3.4.4 Neighbor Discovery operation over IEEE 802.15.4

The first step in Neighbor Discovery is to send a multicast router solicitation including the SLLAO. The size of this router solicitation is of 64 bytes (without compression). Note that SLLAO has a size of 10 bytes but should be aligned to a multiple of 8 bytes [Nar+07]. Using 6LoWPAN adaptation layer [HT11], the IPv6 header can be reduced to 4 bytes as source address can be retrieved from the MAC layer, destination address is set to all-router-link-scope-multicast

(so only the last byte should be carried in-line) and next header should be carried in-line. The remaining fields could be skipped. As a result, multicast router solicitations could be reduced to a size of 28 bytes.

Upon reception, the router sends back a unicast router advertisement including the PIO, 6CO, ABRO and SLLAO. The overall size of such router advertisement is of 144 bytes. Again, such router advertisements could be reduced to a size of 107 bytes thanks to 6LoWPAN (only the next header should be carried in-line). As we can see, such message cannot be transported by an IEEE 802.15.4 frame due to the 102 bytes limitation for data. As a result, this message should be fragmented following the 6LoWPAN specification in two IEEE 802.15.4 frames of 121 bytes and 45 bytes respectively.

Once a router advertisement has been received, the node can configure a tentative global IPv6 address and tries to register this address to the router by sending a neighbor solicitation with the ARO and SLLAO. Such message has a size of 96 bytes. Upon reception, if the claimed address is unique, the router sends back a neighbor advertisement including ARO. This message has a size of 80 bytes. As the IP source of Neighbor Discovery and the IP destination of the corresponding neighbor advertisement should be the address being registered to the router, the IPv6 header can be reduced to a size of 4 bytes with 6LoWPAN (1 additional byte for the context identifier extension and one byte for the next header field). As a result, the size of those messages could be reduced to 60 and 44 bytes respectively.

Class	Type	Size (bytes)
Headers	IPv6 (IPv6_hdr)	40
	ICMP Router Solicitation (RtSol_hdr)	8
	ICMP Router Advertisement (RtAdv_hdr)	16
	ICMP Neighbor Solicitation (NSol_hdr)	24
	ICMP Neighbor Advertisement (NAdv_hdr)	24
Options	Prefix Information Option (PIO)	32
	6LoWPAN Context Option (6CO)	16
	Authoritative Border Router Option (ABRO)	24
	Address Registration Option (ARO)	16
	Source Link-Layer Address Option (SLLAO)	16
Packets	Multicast router solicitation (RtSol)	28 (6LoWPAN)
	Router advertisement (RtAdv)	107 (6LoWPAN)
	Neighbor solicitation (NSol)	60 (6LoWPAN)
	Neighbor advertisement (NAdv)	44 (6LoWPAN)

Table 3.3: Size of headers, options and packets

Trying to fit MIPv6 messages in IEEE 802.15.4 frames is a challenging task. The size of certain messages (i.e. unicast router advertisement), even using 6LoWPAN compression, exceeds the maximum size of a IEEE 802.15.4 frame (102 bytes) and thus must be fragmented (e.g. unicast router advertisement has a size of 107 bytes and must be fragmented in 2 IEEE 802.15.4 frames, of 122 bytes and 45 bytes respectively). In the next section, we will evaluate how IPv6 address auto-configuration is affected by transmitting ND packets in IEEE 802.15.4 frames.

3.4.5 Results of theoretical analysis

To evaluate Neighbor Discovery's performance in the context of WSN, we propose a theoretical analysis using the several equations defined in the previous sections. Values for each parameter are presented in Tables 3.2 and 3.3. In the following, we consider a scenario in which a route-over topology is built. So nodes are directly connected to routers, being 6LR or 6LBR.

Figure 3.6 presents the delays required to perform each stage of the IPv6 stateless auto-configuration procedure in an error free environment. Results are obtained from Equation 3.3 (for results referred to as *min*) and Equation 3.4 (for results referred to as *max*) respectively. As

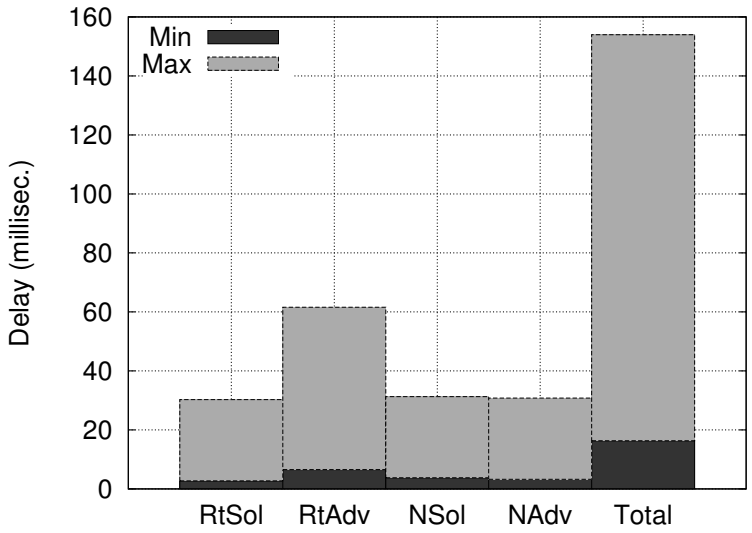


Figure 3.6: Auto-configuration delay without retransmissions

we can see, the delays required for the transmission of router solicitation, neighbor solicitation and neighbor advertisement are below 40ms, even in the case of a busy medium in which the transmitter experiences the maximal number of backoff periods. However, we can see that the transmission of the router advertisement takes more time (up to 62ms) due to the size of such message. Even with 6LoWPAN compression, router advertisements are fragmented in two messages of 121 and 45 bytes respectively. Fragmentation requires more processing from the original sender and the final destination. Also, the successful transmission of two fragments is more subject to errors because one lost fragment triggers the drop of the other fragment, restarting the communication from the beginning. The total delay required to complete stateless auto-configuration is comprised in the interval [16ms,154ms]. Such values are inline with experimental results obtained in [Mon+14] in which the stateless address auto-configuration procedure requires 70ms in average in a similar environment.

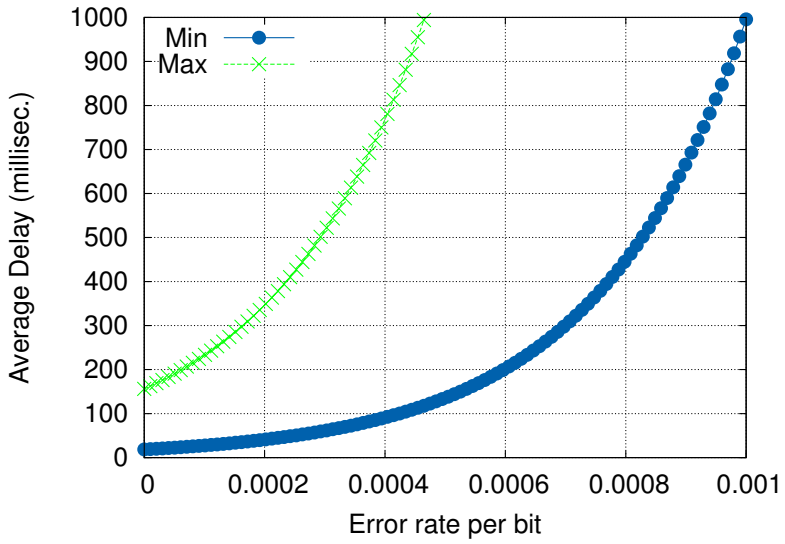


Figure 3.7: Auto-configuration delay regarding link error rate

Figure 3.7 presents the overall average delay required to perform IPv6 stateless address configuration in presence of transmission errors. The X-axis represents the bit error rate (E_r in Equation 3.5). Results are obtained from Equation 3.7 (min) and Equation 3.8 (max). As we can see, the average delay significantly increases along with the bit error rate. In real world deployments, the bit error rate is generally lower than 0.6% [Pet+06]. According to the number of retransmissions allowed by the MAC layer, the delay experienced by nodes can further increase as Neighbor Discovery allows nodes to retransmit up to three router solicitations, each separated by 10 sec, before switching to larger retransmission values. In conclusion, such long delays could not be distinguished from failure and may lead to severe underachievement. Long packets being more subject to transmission errors, this mandates to compress even more neighbor discovery options, especially those transported by router advertisements.

To measure the energy consumption induced by the stateless auto-configuration, we used the values provided by the CC2420 radio component and MSP430 processor data-sheets. Those components are very common and are included in large variety of motes, including TelosB. The energy consumption of this component is illustrated in Table 3.2. In order to convert the times calculated in Equation 3.3 and Equation 3.4 in consumed energy (Joules), we used a linear model presented in [Dun+07] which uses the following equation:

$$energy = \left(\sum_{i}^{components} Current_i \times time_i \right) \times Voltage \tag{3.9}$$

Figure 3.8 illustrates the energy depletion of host and router batteries, each equipped with two 1.5V AA cells while performing stateless auto-configuration. The X-axis indicates the time (in log scale) and the Y-axis the remaining energy (in %) of the node, being host or router. Furthermore, we only represent the results from Equation 3.3, as results for Equation 3.4 are very close (the node can turn off its radio during backoff periods). Note that results corresponding to the node and the router with a single node overlap each other. We can see that most of the energy is consumed at node bootstrap, as the node needs to discover available routers by exchanging router solicitations and advertisements. As previously mentioned, the fragmentation of router advertisements is the main cause of energy depletion of that phase. Once registered, the node should renew its address to the router by exchanging neighbor solicitation and advertisement right before the expiration of its lifetime. For this, we used the minimum allowed lifetime of 60 sec.

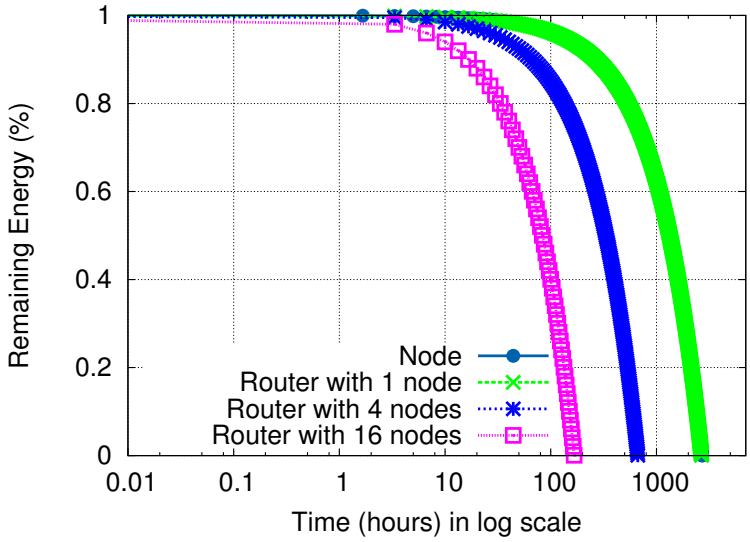


Figure 3.8: energy consumption with two 1.5V AA cells (%)

In Figure 3.8, upon the first registration, the node turns off its radio. Resulting energy depletion is therefore only due to the processor being in sleep mode. After 60 sec, the node wakes up its radio to renew its address to the router and so on. As a result, a host only performing IPv6 stateless address auto-configuration with minimum registration lifetime can last for approximately 2709 hours ($\simeq$ 112 days) before experiencing power outage. This result seems satisfactory but represents the upper bound of node's lifetime because it operates in an error free environment and no other traffic is exchanged. In real conditions, such lifetime is likely to be drastically reduced due to overhearings, collisions, medium contention and any other transmissions that are required by other protocols or by the application that runs on the node. Nevertheless, we can increase node duration by using greater lifetime values for address registration, at the expense of reactivity. By contrast, a router managing 16 hosts only lasts for approximately 166 hours ($\simeq$ 6 days). Such result is far from satisfactory as the main reason of WSN is to exchange data for long period of time. Whenever a router experiences power outage, all nodes (being hosts or routers) using that router are disconnected from the network. Obviously, the more a router manages hosts, the faster it consumes its battery, mainly because routers are now involved in all Neighbor Discovery operations. The new address registration procedure requires that routers periodically exchange unicast neighbor solicitations and advertisements with all neighbor nodes. In addition, router advertisements are now sent in unicast, increasing the number of transactions together with the number of neighbor nodes. By contrast, the load of neighbor discovery is distributed in traditional IPv6 networks: multicast router advertisements allow routers to update simultaneously all link-local nodes, duplicate address detection is performed by nodes, etc. By concentrating all operations on routers, neighbor discovery optimizations for WSN are likely to create single points of failure in the network.

3.4.6 Conclusion for IPv6 statless address auto-configuration

In this section we have analyzed the way neighbor discovery handles IPv6 stateless address configuration in WSN. This work constitutes the first contribution of this thesis, and was motivated by two factors. To the best of our knowledge, we are the first to present a theoretical analysis on IPv6 stateless address configuration using neighbor discovery optimizations defined for WSN. This work could therefore serve the research community as a foundation to further investigations and experimentations. Regarding the current research effort, IPv6 seems to be the most promising technology to interconnect WSN to global Internet and stateless address configuration is one of its major features. Second, two conclusions can be drawn from the results presented in Section 3.4.5:

1. most of options transported by router advertisements are too large to fit in IEEE 802.15.4 frame, even using 6LoWPAN compression.

2. routers being involved in all neighbor discovery operations, are involved in a large number of unicast communications that significantly impact their energy consumption (a router managing 16 hosts only lasts for 6 days in our analysis). Also, routers being responsible for all neighbor discovery operations, become single points of failure in the network.

We therefore believe that is still a long way before IPv6 address auto-configuration can be seamlessly integrated in WSN, even using the optimized version for neighbor discovery. The two conclusions question also the future adoption of protocols such as MIPv6 to provide mobility support in WSN, as their performances are in contradiction with goals of WSN (namely energy efficiency). In addition, there is a slow adoption of such solution (e.g. MIPv6), which has been standardized more than 10 years ago without any major implementation, which questions further research.

Even with optimizations proposed in neighbor discovery, MIPv6 would still have a high energy consumption, which makes them unfeasible candidates for WSN mobility management. Therefore, we searched for a protocol specifically designed for WSN, that will allow seamless integration of a mobile node. We strongly believe that RPL, the standardized routing protocol for WSN,

can provide a better solution for mobility management in WSN, as we will see in the following sections. In the following sections, an overview of generic RPL operations is given, after which some existing mobility management solutions are analyzed. They use external unreachability detection mechanism or enhancements of RPL operations to ensure mobility management.

3.5 RPL - Standard IPv6 Routing Protocol for Wireless Sensor Networks

Sending data from nodes with great constraints on memory or energy, such as the ones present in WSNs, presents a challenging task. Interconnection between nodes experience frequent interruptions, due to instability and lossy links, leading to low packet delivery rates. The change of paradigm regarding the flow of information has redirected the exchange of packets from point-to-point to a point-to-multipoint and multipoint-to-point (or convergecast) pattern. Routing packets in such demanding environments has come under the focus of the IETF ROLL working group and has lead to a standardized approach, the IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) [Win+12].

RPL is a distance vector routing protocol, defined for constrained devices that typically have limited processing power, limited memory and limited energy [Win+12]. As they communicate in general wirelessly, the established radio links between nodes experience quality variations over time and have low data rates. These characteristics, along with the change of paradigm regarding message flow, make RPL a suitable routing protocol for WSNs.

Remaining faithful to the layered architecture of IP, RPL operates over any given link-layer topology, regardless if they are lossy or associated with constrained devices (low-power wireless). As so, RPL has been designed with several applications in mind:

- *industrial applications* - such as in agriculture [Che+12].
- *urban applications* - such as smart meters for electric/gas/water consumption which form the Advanced Metering Infrastructure (AMI) [Doh+09].
- *home automation* - such as actuators for light dimming, control of heating valves or sensors for water leaking [Bra+10].
- *building automation* - such as management of heating, ventilation, and air conditioning (HVAC) [Mar+10] which are included in Building Management Systems (BMS).

To serve all applications that might run in a WSN network, RPL builds a routing tree, called Directed Acyclic Graph (DAG), optimized for specific constraints imposed by the application: minimum latency, high reliability or others. When the Directed Acyclic Graph (DAG) is rooted at a single destination (i.e. a single DAG root) it becomes a Destination-Oriented Directed Acyclic Graph (DODAG).

Nodes will exchange control messages in order to build a routing topology as requested by the application.

ICMPv6 Control Packets

Before any data packets can be exchanged between nodes, a routing topology needs to be built. RPL relies on four new ICMPv6 [Con+06] control messages that are exchanged between nodes:

- DIO: DODAG Information Object. This message caries necessary information for a node to discover a RPL instance, learn its configuration parameters, choose a parent and maintain the DODAG.

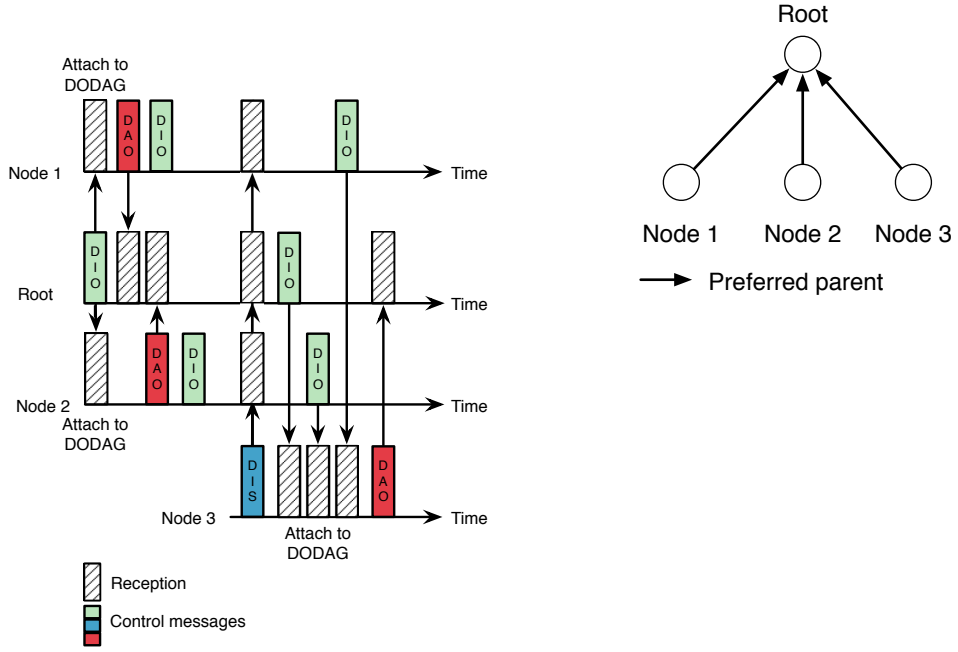
– DIS: DODAG Information Solicitation. Analogue to the Router Solicitation in IPv6 ND, this message may be used to probe the node's neighborhood of potential DODAGs. Nodes receiving a DODAG Information Solicitation (DIS) will reply with a DODAG Information Object (DIO).

– DAO: Destination Advertisement Object. This message propagates destination information upward along the DODAG. Depending on RPL mode, the Destination Advertisement Object (DAO) message can be sent in unicast to the DAO parent (storing mode) or to the DODAG root (non-storing mode).

– DAO-ACK: Destination Advertisement Object Acknowledgment. It is an optional message sent in unicast by a DAO recipient upon an explicit request from the DAO sender or after an error.

Topology Construction

Freeing itself from the limited options in selecting links in a wired environment, in a Wireless Sensor Network environment, RPL, has to discover potential links in the neighborhood and select appropriate peers. Exchanging the above mentioned control messages RPL will build routes following a graph. Built routes are optimized for traffic to/from one or more roots that act as sinks for the topology. This allows RPL to organize a topology as a DAG that is partitioned into one or more Destination-Oriented Directed Acyclic Graphs (DODAGs), one DODAG per sink. A generic message exchange sequence for constructing both upward and downward routes is presented in Figure 3.9a and the resulted DODAG after exchanging control messages in Figure 3.9b. Further, a more in depth view of each route construction is given.



(a) Message sequence during topology construction (b) Resulted DODAG

Figure 3.9: Topology construction

– *DODAG Construction and upward routes.* RPL nodes construct and maintain DODAGs according to an Objective Function (OF) through reception of DIO messages. The OF

present in DIO message, defines RPL route selection and optimization. Also, it helps nodes translate various constraints into a representative value called *rank*, an approximation of the distance from itself and the DODAG root. After processing the DIO message the node will also select a parent according to the advertised OF. The OF contains one or more metrics and constraints. The available metrics are designed to achieve specific goals like throughput, latency or link reliability (e.g. using ETX), as defined by [JP.12].

From the OF, a node builds a set of neighboring nodes, which can be reached through link-local multicast. Then, the node chooses a subset of nodes from the neighboring set to form the parent set. This final pool will give the preferred parent, the preferred next hop when packets are sent upward to the root.

Sending DIO messages during DODAG construction is governed by the trickle algorithm, which will reduce control traffic as all nodes join the DODAG and the topology becomes stable. In the next section more details about the inner working of the trickle algorithm will be given. After some time the network topology becomes stable and if a new node enters the network (e.g. Node 3 in Figure 3.9a), a new DIO message may arrive after a long time. Sending a DIS message in multicast will allow the requesting node to attach faster to the DODAG, as neighboring nodes will reply with DIO messages back (e.g. as seen in Figure 3.9a).

– *Destination advertisement and downward routes.* RPL can construct downward routes using DAO messages. DAO messages are optional when applications require point-to-multipoint or point-to-point traffic [Win+12]. RPL offers two Mode of Operation (MOP) for downward traffic - storing, or fully statefull and non-storing, or fully source routed, only one mode used at a time in a DODAG. RPL does not allow mixing storing and non-storing modes in the same network. In [Ko+14] has been shown that this can cause packets to bounce between nodes that have different MOP. Packets exchanged between two nodes in a point-to-point connection travel all the way up to the root in non-storing more or up to a common ancestors in storing mode and than back down to the final destination. The root can also take advantage of the point-to-multipoint connection to send data towards other nodes in the DODAG.

  Nodes choose a next hop destination for the DAO message, which becomes the DAO parent (as seen in Figure 3.9a). Depending on the configured MOP, nodes should be sufficiently provisioned: in storing mode each node must be able to maintain a routing table for all child nodes in his sub-DODAG and in non-storing mode, the root must maintain source routing information learned from all received DAO messages. Lifetime considerations dictate the periodicity of DAO generation, the default value in RPL being 60 sec. Changes in topology, such as a new point of attachment after changing parents will generate new DAO messages.

The Trickle Algorithm

As we have seen, during the DODAG construction, DIO messages are sent using the trickle algorithm [Lev+11]. The trickle algorithm can double the time interval between two consecutive DIO messages or reset it to the minimum allowed value, depending on the frequency of changes in the topology. The trickle algorithm can be reset based on received messages or whether an inconsistency has occurred. An inconsistency is considered when the node receives a DIS message in multicast or when the node joins a new DODAG (e.g. by receiving a DIO message from neighboring nodes, which indicates a different DODAG that the one it was attached previously). These events, but also others that can be included depending on implementation, will bring the trickle timer to $I_{\min} = 2^{DIOIntervalMin} ms$ (by default the RPL standard defines the minimum interval between two DIO messages as $DIOIntervalMin = 3$ which gives $I_{\min} = 8ms$). When no inconsistencies are detected until the end of the current trickle timer interval, the interval value will double its value and a DIO message is transmitted in the neighborhood. Intervals can double up to $I_{\max}$ doublings of $I_{\min}$ interval (by default there are 20 doublings, leading to $I_{\max} = 2.3h$).

The pattern of DIO message transmission, shaped by the trickle algorithm, is meant to decrease the control traffic overhead as the neighborhood of the node becomes stable and the topology completes the initial setup phase. However, any inconsistency detected by a node will reset its trickle timer and trigger the sending of new DIO messages. This should quickly mitigate any change in the topology and re-establish connectivity between nodes in the DODAG.

Leaf node in DODAG

A node may attach to the DODAG as a leaf when the node does not fully support the advertised OF. When a node attaches as leaf, it will no longer extend the connectivity of the DODAG, as it will not advertise itself as a router (suppression of DIO message transmission). A node in leaf node must follow rules set in the standard [Win+12].

- Any DIO message advertised must contain INFINITE rank.
- It may suppress DIO message transmissions, but must reply to any DIS messages in unicast.
- It may transmit DAO messages under conditions defined by [Win+12].

Loop Avoidance and Detection

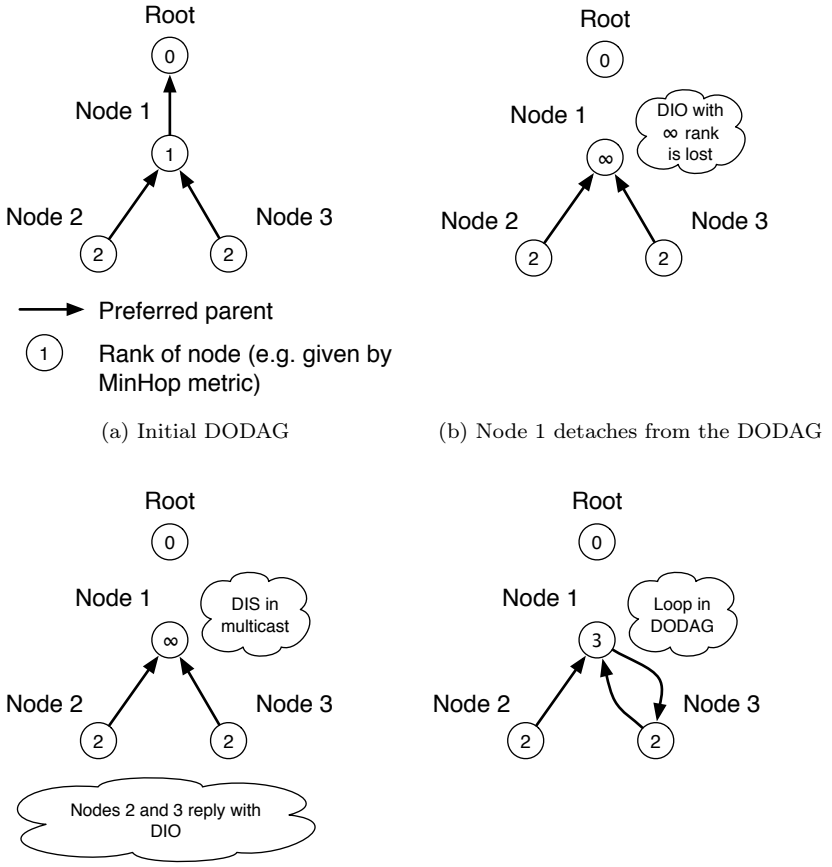
Maintaining a tree structure is important in RPL, but changes in topology can create loops. RPL tries to avoid loops when topology changes occur and includes loop detection capabilities through rank-based data-path validation mechanisms. When implemented, RPL will not be able to guarantee loop-free path selection or set an upper limit to the convergence delay time, but as soon as paths with loops are used, it can detect and repair the loop.

Nodes in a DODAG can become part of a loop if they detach from the DODAG and reattach to a node in its previous sub-DODAG. This can happen when a node makes a local repair. During a local repair, the node will remove all parents from the parent set, announce its disconnection from the DODAG with a DIO message advertising INFINITE_RANK and send periodic DIS messages in multicast until new DIO messages are received. Even though it advertises INFINITE_RANK, because of local conditions in WSN packets can be lost and the node may end up reattaching to its prior sub-DODAG, as seen in the stages of loop creation from Figure 3.10.

IPv6 address auto-configuration for RPL

RPL constructs a DODAG using dedicated control messages. As nodes join the DODAG, they choose a preferred parent and can trigger an address allocation process, even before the whole topology is constructed. Authors in [Zhu+13] submitted a draft to the IETF to propose an IPv6 address auto-configuration process for RPL. The address allocation process consists of two phases:

- *Address allocation*: Nodes in the network send DIO messages with a dedicated option which contains the IPv6 address of the node. Any receiving neighbor, that becomes the child of this node, will process this DIO message and compute its own address. The child takes the IPv6 address in the received DIO message and considers it as a prefix. It then generates a random n-bit string and concatenates the received prefix with the random string, obtaining a new IPv6 address (e.g. received parent address is "0011", a random 4 bit string is "0101", then the address of the child will be "00110101"). The child, after generating an address, will send back to the parent this new address in a DAO message (with an added option to transport the new address). The child waits for a DAO-ACK message from the parent (with APPROVE or REJECT option in the DAO-ACK message). If the child receives a DAO-ACK message with an APPROVE option, this will confirm the child address and it can advertise the address in following DIO messages. If a REJECT option is present in the DAO-ACK message, the child will restart the process of choosing a new address.



(a) Initial DODAG

(b) Node 1 detaches from the DODAG

(c) Node 1 searches a new attachment point

(d) Node 1 reconnects to a node from its previous sub-DODAG

Figure 3.10: Loop formation in DODAG

– *Address conflict detection*: Parents of nodes receive DAO messages from his child nodes with their chosen address and checks in the Address Lease Table (a table that contains leased addresses to nodes in the node's sub-DODAG) if the address have been already leased. If no conflicts are found, the parent sends a DAO-ACK message with the APPROVE option and adds the child's address in the Address Lease Table. If the parent has already leased the child's address to another node, it will send a DAO-ACK message with the REJECT option.

These two phases constitute the basis of address auto-configuration in RPL. Addresses can be reused in the sub-DODAG of a parent. When nodes leave the sub-DODAG, they should notify their parent with a no-path DAO. The no-path DAO notifies the parent that a child leaves its sub-DODAG, so the parent can free the leased address from the Address Lease Table and delete any downward route associated to the child node. If the child node does not send a no-path DAO message when it leaves, the parent monitors the traffic received from the child node. If no traffic is received from the child node, the parent can query (with unspecified messages) the child node and remove the lease address if no reply is received.

Using address auto-configuration for RPL we should be able to solve the above-mentioned problems of IPv6 address auto-configuration using neighbor discovery. Integrating address auto-configuration in RPL can be done with minimal overhead, by attaching options to already exchanged control messages like DIO, DAO or DAO-ACK and keeping the standard RPL operation.

In the next section we will go through the built-in mechanism in RPL that can be used to mitigate slow changes in the topology and enables nodes to re-establish connectivity. After this overview, we will concentrate our attention to other approaches that help integrate a mobile node in a RPL network.

3.5.1 Analysis of RPL under mobility

In wireless environments, such as the one present in WSN, links are lossy as nodes have limited transmission and energy capabilities. All of this may generate network instability, with intermittent node disconnection and/or disappearance that may perturb RPL normal operation [Iov+13]. In addition, a large variety of applications, ranging from target tracking to wildlife monitoring ([All+07], [Leg+08], [Sik+06]), require the support of node mobility. However, mobility will add further challenges as now, communication will not only be impacted by perturbations on the wireless channel, but also from the physical movement of nodes. The network can therefore be logically or physically partitioned.

RPL has a build-in mechanism that can mitigate mobility, by allowing nodes to change their preferred parent in order to reconnect to the graph. The reconnection occurs when a node receives a DIO message advertising a better rank than the one of its current preferred parent. However, this mechanism does not always give a node the possibility to change its preferred parent. This happens when a node can no longer reach its preferred parent and all following DIO messages advertise a worse/higher rank than the one of the preferred parent, thus the node remains disconnected from the graph. Such situation is illustrated on Figure 3.11, in which a node remains disconnected from the graph because its preferred parent (Parent 1) is no longer available and all other parents in the parent set (Parent 2) advertise a rank that does not determine a change of parent (e.g. Parent 2 can have a better/worse rank with a $\pm\delta$ variation, but even with a better rank, due to stability considerations in RPL - namely a tolerated variation of the rank, the node will not change Parent 1 as preferred parent). As a result, such situation is likely to increase contention on the medium, the energy consumption and packet loss.

Managing the parent set is not specified in RPL, namely why and when a node should be removed from the parent set. When mobility is detected, finding a new parent by managing the parent set will allow faster re-connection to the DODAG for the mobile node. There are several ways to achieve a better management of the parent set, which will improve RPL mobility support:

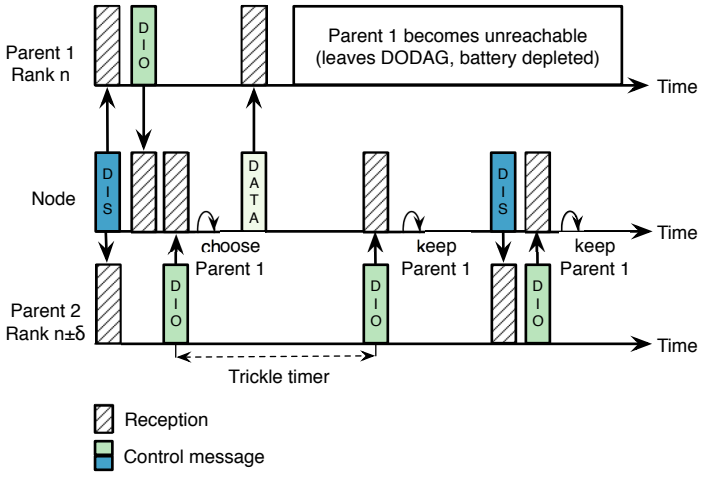


Figure 3.11: RPL parent management

– In [Win+12], RPL suggests using external unreachability detection mechanisms, as we will seen in Section 3.5.2.

– The research community has proposed several state of the art solutions, modifying an enhancing RPL operations to enable mobility management, presented in Section 3.5.3.

All these mechanism can be used to achieve the goal of seamless integration of a mobile node in the RPL network.

3.5.2 Unreachability detection mechanisms suggested by RPL

Determining the unreachability of a node is suggested in RPL to be done via external mechanisms such as Neighbor Unreachability Detection (Neighbor Unreachability Detection (NUD)) [Nar+07], Bidirectional Forwarding Detection (BFD) [KW10] and hints from lower layers via Layer 2 (L2) triggers [Ter+08]. All those mechanisms have an internal process that indicates when a remote node is unreachable, allowing a node to start searching for a new RPL parent. The node will first search a suitable candidate in its RPL parent set and if there are no parents available, it will do a local repair in RPL. Continuous communication on transient links, where losses are recurrent, should be achieved when one of those mechanisms is coupled to RPL. The inner workings of all mechanism when unreachability is detected are detailed below.

Neighbor Unreachability Detection

Neighbor Unreachability Detection (NUD) is part of Neighbor Discovery for IPv6 (ND) [Nar+07]. It tracks all paths between active neighboring nodes and specifies when a neighbor is unreachable. The state of connectivity between neighbors is stored locally on each node in a structure called neighbor cache.

NUD enables neighbors to exchange Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages to confirm reachability. Each neighbor has an entry in the neighbor cache for all connections it has with other nodes in the same network. Cached values for nodes can be:

– REACHABLE - communication is granted between nodes,

– STALE - the neighbor is no longer known to be reachable but no action is taken until traffic is sent to this neighbor,

– DELAY - optimized state that delays sending probe for *DELAY_FIRST_PROBE_TIME* seconds (node waits for reachability confirmation from upper layers) and

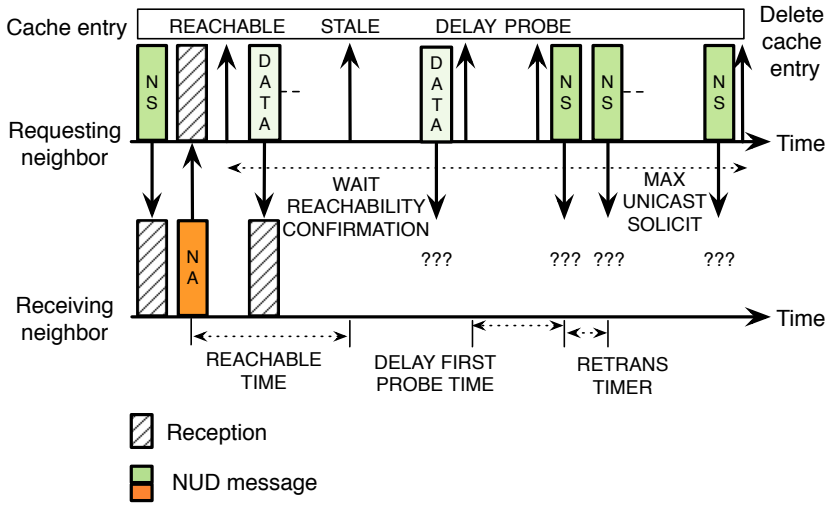


Figure 3.12: NUD message exchange

– PROBE - NS are sent until reachability is confirmed or the maximum allowed number of probes (*MAX_UNICAST_SOLICIT*) is sent.

Timers, which are illustrated in Figure 3.12, manage the exchange of control messages and trigger the removal of the cache entry. Default values for timers give a 30 sec reachable time window. After this time elapses and a data packet needs to be sent, a probe is sent with a default 5 sec delay (*DELAY_FIRST_PROBE_TIME*) and than each second until *MAX_UNICAST_SOLICIT* probes are sent (3 retransmissions by default). In the worst case and considering the timer default values, it takes 38 sec for NUD to detect the unreachability of a neighbor.

Bidirectional Forwarding Detection

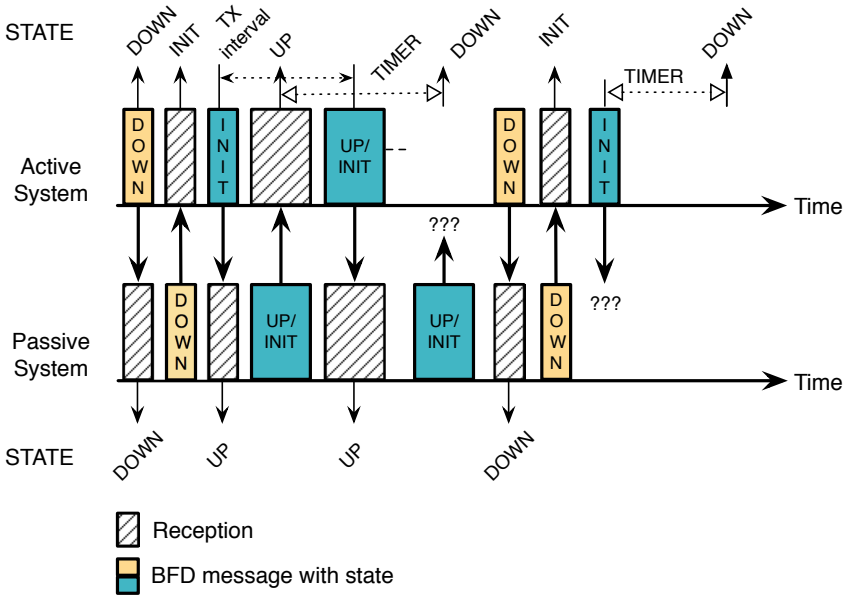


Figure 3.13: BFD message exchange in asynchronous mode

Bidirectional Forwarding Detection (BFD) [KW10] is a simple Hello protocol that detects failures in communication with a forwarding plane next hop. A pair of nodes (called systems) exchanges BFD messages encapsulated in UDP packets to maintain reachability information. The path between two nodes is declared operational when two-way communication can be established. Systems exchange BFD messages and follow a state machine that brings the states of the systems in UP state. Once in UP state, the systems will declare the communication established. When no messages are received for long enough, BFD considers that the neighboring system has failed. Operation of BFD in asynchronous mode, along with the state of each system, is presented in Figure 3.13. BFD messages are sent by default every 20 sec and if one packet is lost, the system declares the state *DOWN*.

Hints from lower layers via Layer 2 (L2) triggers

Although specific to each MAC layer, the hints from lower layers via Layer 2 (L2) triggers [Ter+08] share common structure in the form of L2 Abstractions. Services between layers are provided in the form of primitives, which enable synchronous communication between layers. Two pairs of primitives are defined to be used when events occur: Request/Confirm and Indication/Response.

Primitives can be used in three different cases based on their types:

- *Type 1*: Provide information to upper layers; information is provided immediately to upper layer through a request-confirm message exchange;
- *Type 2*: Notify upper layer of L2 events asynchronously, indicating each occurrence of registered events to upper layers;
- *Type 3*: Control L2 actions from upper layers; Request primitive is used to interact with lower layer which will reply with Ack or Nack in a Confirmation primitive.

As we can see, this mechanism offers only a generic framework that should be adapted to the specific MAC protocol in use. It offers an increased reactivity to layer 2 (MAC) events that can be used by upper layer protocols.

In Section 5.1 we will present how these mechanisms can be integrate with RPL to provide mobility support. In the following section we present enhancements of RPL operations to support mobility management. The mobility management is achieved through efficient management of the RPL parent set at the mobile node.

3.5.3 Enhancements of RPL operations to support mobility management

More recently, researchers have focused their attention to provide mobility management solutions, created specifically for WSNs. In this section, we will focus on solutions that extend or modify RPL to manage the mobility of moving nodes.

The authors of [Lee+12] propose to remove the trickle algorithm for DIO messages and replace it with a fixed value ranging from $2sec$ to $10sec$. Obviously, the DIO message overhead is proportional to the DIO period - the lower the DIO period, the higher the DIO message overhead. However, lower DIO periods allow the reduction of disconnection time, as new DIO messages are processed more frequently. However, a fixed sending rate will generate constant DIO messages in the whole network, leading to extra control traffic even in areas not crossed by mobile nodes. Such growth in control traffic may significantly increase the energy consumption together with the contention on the wireless medium. In addition, this proposal relies on the RPL built-in method to change parents. As presented previously, the reception of new DIO messages does not necessarily trigger a parent change, even if the current parent is unreachable. Nevertheless, the authors mitigate such situation by using ETX, as this metric is dynamic enough to delimitate two parents. Their simulation with 10 mobile nodes moving one after another in a line, with an

DAG root in the middle of the area show that nodes in the front experience larger disconnection periods than nodes further back. When the first mobile node arrives in the range of the root, it will connect to the DODAG and start propagating the information to the mobile nodes in the back. Thus, mobile nodes further back can connect faster to the DODAG and do not need to wait until they arrive in the range of the root. Also, the authors acknowledge the increase of DIO message overhead with the increase of DIO period and also the increase of DIO message overhead with the increase of mobile node speed, as the changes in topology become more frequent and are reported with new DIO messages.

In [Kor+12], the authors propose a dynamic DIS management procedure. In a nutshell, they send DIS messages with different intervals to force the refresh of routing information. When a mobile node experiences high mobility (i.e. it changes several times its preferred parent), the interval between two consecutive DIS messages should be small in order to allow for fast route informations update. By contrast, if a mobile node remains connected to the same parent, the interval between two consecutive DIS messages should be large because the node is not moving or moving slowly. For this, they define $DownDIS$ which is the number of parents change above which the inter-DIS interval should be divided by 2, $UpDIS$ which is the number of times a mobile node chooses the same parent above which the inter-DIS interval should be multiplied by 2, and I_{min} and I_{max} which are respectively the minimum and maximum time interval between sent DIS messages. Their proposal can be summarized in the algorithm 3.14, where $preferred\_parent\_changed$ is the number of changes of the preferred parent in the current inter-DIS interval.

```
1: dag ← Current_DAG
2: next_dis ← next_dis+1
3: if preferred_parent_changed = 1 then
4:    /* Increase number of preferred_parent changes */
5:    number_parent_changed ← number_parent_changed + 1
6: else
7:    number_same_parent ← number_same_parent+1
8: end if
9: if preferred_parent_changed = 1 then
10:   if periodic_dis ≥ 2 × I_min and number_parent_changed ≥ DownDIS then
11:      DIS_period ← DIS_period/2
12:      number_parent_changed ← 0
13:   end if
14: else
15:   if periodic_dis ≤ RPL_DIS_INTERVAL/2 and number_parent_changed ≥
      UpDIS then
16:      DIS_period ← DIS_period×2
17:      number_same_parent ← 0
18:   end if
19: end if
```

Figure 3.14: Dynamic DIS management

This solution is interesting as it only involves nodes close to mobile nodes and not the whole network. However, the transmission of a DIS message may unnecessarily reset the trickle timer of all nodes in range (i.e. starting again with frequent DIO messages) and trigger the transmission of multiple DIO messages while the mobile node is still connected. Furthermore, this proposal also relies on RPL build-in procedure to change parents. The situation in which a mobile node does not process new DIO messages due to the rank value is also mitigated by the use of ETX. According to their results, they suggest to set $DownDIS = 1$, $UpDIS = 5$, $I_{min} = 3s$ and $I_{max} = 60sec$ to achieve the maximum packet delivery ratio. However, they do not specify the default initial value for inter-DIS interval, I_{init}, to start the algorithm.

Another solution to provide mobility support in RPL is Co-RPL [Gad+14], an extension of RPL that keeps track of the position of a mobile node while moving to improve network performances. The described network is composed only of static roots and mobile nodes. The position of the mobile node is computed relative to coronas [ZS09], circular region with a certain radius centered at the DAG root. There can be several roots in the network, each constructing a DAG. Authors set the radius of each corona to the maximum transmission range of each sensor and coronas are identified by an ID. Nodes can belong only to one corona at a time (even if coronas overlap), but they can switch between coronas. DIO messages are sent periodically by the DAG root in this proposal (at an unspecified rate), so that DAG root is aware of the position of the nodes in real time. Nodes operating with this solution send modified DIS and DIO messages to distinguish themselves from other nodes using standard RPL (set the MSB in the flags field of DIO and DIS messages and add corona information - corona ID in DIO messages). Nodes also maintain a neighboring table that is updated after each new received DIO message, with information about IDs of DAG, node or corona and the link quality (which the authors do not specify how they obtain). The proposed solution has three mechanisms that work together to maintain connectivity for mobile nodes:

1. A corona mechanism, where nodes compute their corona ID from received DIO messages. The root sends the first DIO message with the corona ID of 0, then, each node that receives a DIO message will check it's neighbor table, select the minimum value for corona ID of the available neighbors and increment by one the value. Than, the node will broadcast its corona ID in a new DIO message.

2. Different operation for the DAG root and for the mobile nodes. The DAG root periodically sends DIO messages with the corona ID 0 (at an unspecified rate); if the DAG root receives a DIS message, it will immediately send a DIO message, regardless of the time left until the next scheduled DIO message. The mobile node listens for DIO messages and once it receives the first DIO message, it will join the DAG and compute its rank (the used metric is not clearly stated, but it will give the quality of neighbors) and corona ID. Once multiple DIO messages are received, the mobile node will choose a preferred parent will the minimum corona ID and best available quality. If the mobile node does not receive any DIO message, it will send periodically DIS messages (with an undefined frequency) until a DIO message is received. When the node moves inside the same corona, it will be aware of this as the list of neighbors from its table will necessarily change with the change of position (the quality of the neighbor will change).

3. A path recovery mechanism that is used when a mobile node cannot forward data packets to the next-hop node, it backwards the data packet to any node in the high corona level and notifies its sub-DODAG not to send any data by sending a DIS message. The mechanism to detect the next-hop unreachability is not detailed by the authors.

Authors run a simulation campaign where they compare their solution with standard RPL. They improve packet loss by 45% and lower energy consumption by 50% when compared to standard RPL. This evaluation can be valuable for networks where all nodes are mobile and only the root is static.

Authors of [Fot+15] develop a mobility mechanism for RPL, mRPL, by integrating a mobility detection mechanism (smartHOP [Fot+12]), based on received RSSI levels, in RPL. The authors adapted the smartHOP mechanism to use RPL control messages. With smartHOP in RPL, the mobile node, once connected to a parent, will send a number of data packets to the parent, after which the parent will send a unicast DIO message in reply to data packets. The received unicast DIO message contains the average RSSI level and filters implicitly asymmetric links. As long as the received RSSI levels are above a threshold, data transmission continues. When the mobile node detects that the RSSI value drops under a threshold, it will start searching for a new parent, while continuing the communication with the current one. Finding a new parent for the mobile node is done through a burst of DIS messages in broadcast (but their number is not specified).

The receiving nodes will reply with DIO messages in unicast, delaying their reply in such a way that collisions do not occur at the mobile node. This process (sending a burst of DIS messages from the mobile node and receiving unicast DIO message) continues until the mobile node finds a new parent with a high quality link (the received RSSI above a threshold) and resumes data transmission with that parent. After a successful handover, the previous parent is removed from the parent set. An overview of mRPL operations can be viewed in Figure 3.15.

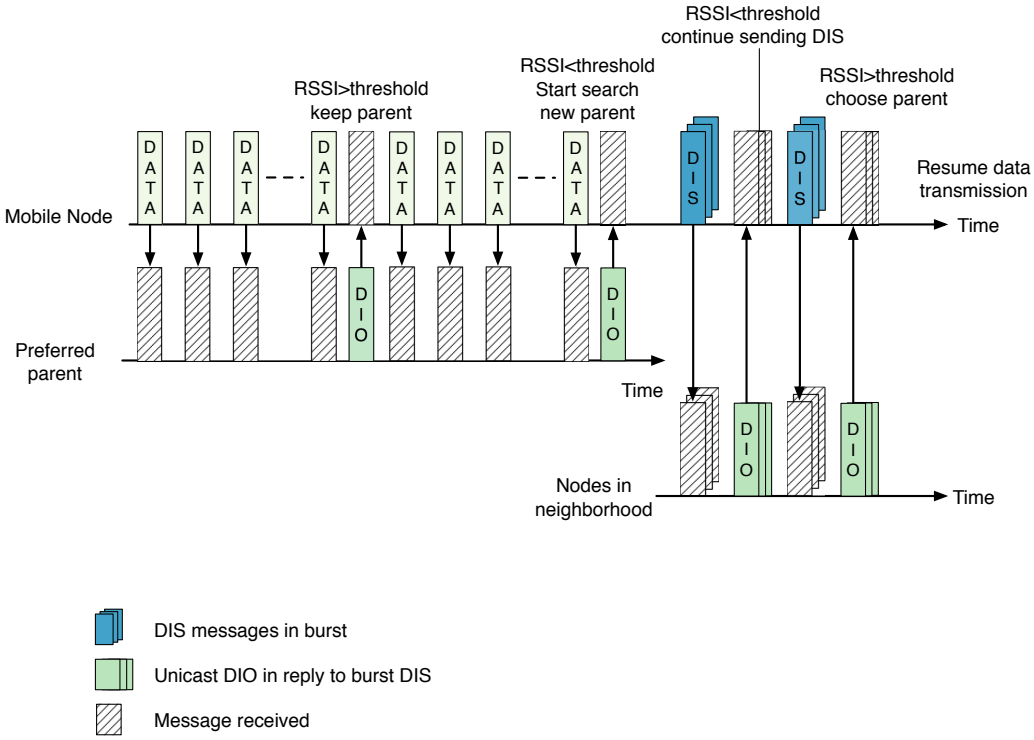


Figure 3.15: mRPL mobility management

After a simulation and experimentation campaign, where authors compared the build-in parent selection mechanism with mRPL, the authors conclude that they improve mobility management in RPL in several areas: high packet delivery, responsiveness to network dynamics (via increased number of DAO messages sent by the mobile node as a measure of successful parent change), effectiveness at high data transmission rate. The proposed solution needs high data rate to maintain the connectivity of mobile nodes (the packet delivery date drops by 24% if data rate is increased from 100 ms to 5 sec), but regular applications in WSN need far less traffic (e.g. 1 packet/15 sec for vehicle tracking applications [KK08]). While the results are encouraging, the limited simulation time (2 minutes) does not reflect a realistic interpretation of processes that happen in a RPL network. The network needs time to arrive in a stable state [Iov+13], so, in our opinion, with such short time the solution does not provide sufficient insights on long term performances. In addition, setting thresholds based on received RSSI does not give reliable results, as [CT10] show and can lead to increased handovers. However, this analysis is difficult to make given the short simulation period (the duration of experiments is not revealed by the authors in their work). Lastly, the energy consumption with such a high traffic rate in the network is elided, even though simulations and experiments have been done.

All these solutions rely on the built-in method in RPL for changing the preferred parent, namely the reception of a new DIO message with a better rank. Authors try to speed up the process by either sending DIO messages periodically like in [Lee+12] and [Gad+14] or by tuning the inter-DIS interval in such a way that nodes receive updated information from the neighborhood like authors of [Kor+12] propose. In [Fot+15], authors propose a new way of

sending DIO and DIS messages based on thresholds of received RSSI and rely on high data rate to achieve seamless handovers.

3.6 Conclusion

In this chapter, we have made an overview of mobility management at the network layer. We started looking at MIPv6 as a solution to mobility management in WSN. One of the main difficulties found by the research community with MIPv6 is the long delay before mobility is detected. This problem has a root in the use of the neighbor discovery protocol for standard IP networks that lacks optimizations for WSN. In our first contribution, we have made an analysis of the latest optimization of neighbor discovery for WSN from a theoretical point of view to determine if the optimized ND protocol can speed up the movement detection in MIPv6. We found out that the size of router advertisements with IPv6 address auto-configuration options enabled exceeds the IEEE 802.15.4 frame size and that routers involved in neighbor discovery process have higher energy consumption and become single point of failures in the network. This proves that further adoption of MIPv6, as a mobility management solution for WSN, is not feasible. That is why, we shifted our focus to a solution developed from ground up for WSN, namely IPv6 Routing Protocol for Low-power and Lossy Networks (RPL). The routing protocol is in our opinion better suited to manage mobility than is MIPv6 in WSN. Solutions using external unreachability detection mechanisms or enhancing RPL operations for mobility management are envisaged. In the next chapter, we present our second contribution, an enhancement of RPL operations for mobility management, which is compared through an extended simulation campaign with other state of the art proposals.

Chapter 4

Enhancements of RPL operations for mobility management

We have seen previously that MIPv6 is not a feasible solution for mobility management in WSNs and that using a routing protocol specifically designed for WSNs, the IPv6 Routing Protocol for Low-power and Lossy Networks can be a solution. Since the build-in mechanisms that can mitigate mobility, by allowing a mobile node to change the preferred parent, can leave a node disconnected for a long period, several solutions to enhance RPL operations have been proposed. In this chapter, we present our contribution to support mobile nodes in RPL by enhancing RPL operations. We propose that only nodes with which the mobile node communicates change their behavior. Our solution does not affect the configuration of the rest of the network. Furthermore, new operations include a reverse trickle timer, which will allow fast re-attachment to the graph after the mobile node changes its attachment point. Mobile nodes that exchange RPL control messages should have minimum impact on the connectivity or stability of the DODAG. Selecting a mobile node as a preferred parent can generate loops in the network [Kor+12]. We therefore advocate that mobile nodes only connect as leaves in the DODAG. Such limitation is already included in the specifications of RPL. We consider, similarly to [Kor+12], that the mobile nodes are identified and advertise their mobility status in control message. For this, we choose to modify the DAO message send by mobile nodes, by introducing a Mobility Flag (MF) in the flags field of DAO messages. The DODAG we build supports upward and downward routes (in non-storing mode of operation (MOP)). We will start presenting the reverse trickle algorithm, that will govern the timing of DIO messages sent by the parent of the mobile node to the mobile node. We then continue with an enhanced RPL message sequence, that will be carried out only between the mobile node and its parent and will allows to manage the mobility of the mobile node only with the help of RPL control messages. A thorough simulation campaign of our second contribution, done alongside two other state of the art solutions that enhance RPL mobility management capabilities shows what improvements our contribution brings. Conclusions of our second contribution will end and round up this chapter.

Contribution

We propose in this chapter the following enhancements to RPL operations for mobility management:

1. The reverse trickle timer, that governs the sending of DIO messages from preferred parents of mobile nodes.
2. The Mobility Flag introduced in DAO messages sent by mobile nodes, which helps parents of such nodes to identify the presence of a mobile node in their sub-DODAG.
3. New sequence of RPL messages that enhance the capabilities of RPL to accommodate mobile nodes along static nodes in the same network.
4. Evaluation through simulation alongside two state of the art proposals from the literature.

4.1 The reverse trickle timer

The standard trickle algorithm in RPL starts with a short interval between two consecutive DIO messages, in order to quickly react upon the various topology changes that occur when the graph is built. As the network become stable [Iov+13], the interval between DIO messages increases in order to limit the control traffic overhead. To support mobile nodes, we propose a reverse trickle timer that starts from a maximum allowed value between DIO messages and halves the sending intervals after each new DIO message sent. Let I_{min} and I_{max} denote the minimum and maximum interval between two consecutive DIO messages. For a mobile node that moves in the network and connects to a parent, it is very likely that it will remain connected to this parent for a long time. The mobile node will connect to a parent when it enters the parent's range and, depending on speed, it will move for some time inside the range of the parent, before leaving the neighborhood. We advocate that the interval between DIO messages should be large, as no change is expected in the next period, so it is set to a value close to I_{max}. Then, the more the mobile node spends time connected to the same parent, the more it is likely to move outside the coverage of the parent. Therefore, after each DIO message, the reverse trickle timer algorithm halves the previous interval between DIO messages. This operation continues until the minimum allowed interval between DIO messages, I_{min} is reached. Once the trickle timer reaches I_{min}, the parent requests new DAO messages from any connected node. In [Win+12] a node can request new DAO messages from the children nodes in its sub-DODAG if it sends a DIO message in which the Destination Advertisement Trigger Sequence Number (DTSN) field is increased. If a new DAO message from the mobile node is received, the algorithm starts again on the parent node. A parent switches from the standard trickle timer to the reverse one whenever a mobile node connects to it. The parent switches back to the standard trickle timer after the mobile node is not anymore in its sub-DODAG. Algorithm 4.1 illustrates this procedure.

4.2 New enhancements of RPL operations for mobility support

When a mobile node enters a DODAG, the mobile node is disconnected at first and will send DIS messages in multicast in order to discover any DODAGs in its neighborhood. Nodes in the neighborhood that receive the DIS message will reset their trickle timer to the minimum

```
1: if DAO_reception && MF = 1 then
2:   mobile_node ← TRUE
3: end if
4: while mobile_node = TRUE do
5:   Itmp ← Imax
6:   while Itmp/2 > Imin do
7:     I ← random(Itmp/2 , Itmp)
8:     Schedule next DIO in I
9:     Itmp ← Itmp/2
10:  end while
11:  I ← random(Imin , Itmp)
12:  Schedule next DIO in I with DTSN+1
13:  if !DAO_reception or DAO_reception && MF ≠ 1 then
14:    mobile_node ← FALSE
15:  end if
16: end while
17: Go back to standard trickle timer
```

Figure 4.1: Reverse trickle timer algorithm

allowed value and will begin to send DIO messages in the neighborhood. Once the mobile node receives DIO messages, it will choose a preferred parent, attaching itself to the DODAG. After the mobile node is attached to the DODAG, it will advertise its address to the root, so that the root can calculate the downward route towards the mobile node. Address advertisement is done by sending DAO messages toward the root with MF flag set to 1. The DAO message is then sent to the preferred parent. When the preferred parent receives the DAO message, it will check if the MF flag is present in the message before forwarding the message upward, toward the root. A parent receiving a DAO message with MF flag set to 0 (i.e. from a static node) continues with default RPL operations for DAO messages, simply forwarding the received DAO message to the root. If the MF flag is set to 1 in the received DAO message, the parent detects that the address belongs to a mobile node and enters the reverse trickle timer algorithm. It will begin sending DIO messages toward the mobile node, in unicast, at the end of each interval defined by the reverse trickle algorithm. The reverse trickle algorithm continues to halve the remaining interval, until the minimum value for the reverse trickle timer is reached (minimum interval between two consecutive DIO messages). When the minimum value for the reverse trickle timer is reached, the parent of the mobile node will send a multicast DIO message with an increased Destination Advertisement Trigger Sequence Number (DTSN). The increased DTSN triggers the sending of a DAO message from any attached node. If the mobile node is still attached to the parent, it will also reply with a new DAO message with MF set to 1. If the parent receives another DAO message with MF flag set to 1, it resets the reverse trickle timer to the maximum value and starts again the algorithm. If the parent receives no DAO message from a mobile node, the parent goes back to the standard trickle timer, continuing from the value that it has reached before the mobile node chose the node as preferred parent. This means that the parent does not have anymore a mobile node in its sub-DODAG.

The mobile node, after it attaches to a parent and sends a DAO message with MF flag set to 1, begins to monitor the received DIO messages from the preferred parent. It will monitor the time since the last received DIO message from its parent. We define a threshold, $Dthresh$ that represents the sum of the intervals between which DIO messages are sent, as given by the reverse trickle algorithm (e.g. if we have $Dthresh = 2$ this will mean that we sum the first and second interval between DIO messages given by the trickle algorithm - I_{max} and $I_{max/2}$). The maximum value for $Dthresh$ is the sum of the maximum number of DIO intervals that can be obtained between I_{max} and I_{min}. For the mobile node, $Dthresh$ represents the maximum interval of time in which it will wait for a DIO message from its parent before considering that the connectivity

is lost. The first DAO message with MF flag set to 1 will trigger the usage of the reverse trickle timer at the parent node. This also means that the mobile node also knows after how much time it is supposed to received the next DIO message from the parent, time given by *Dthresh*. Therefore, after each received DIO message, the mobile node sets a timer to *Dthresh*.

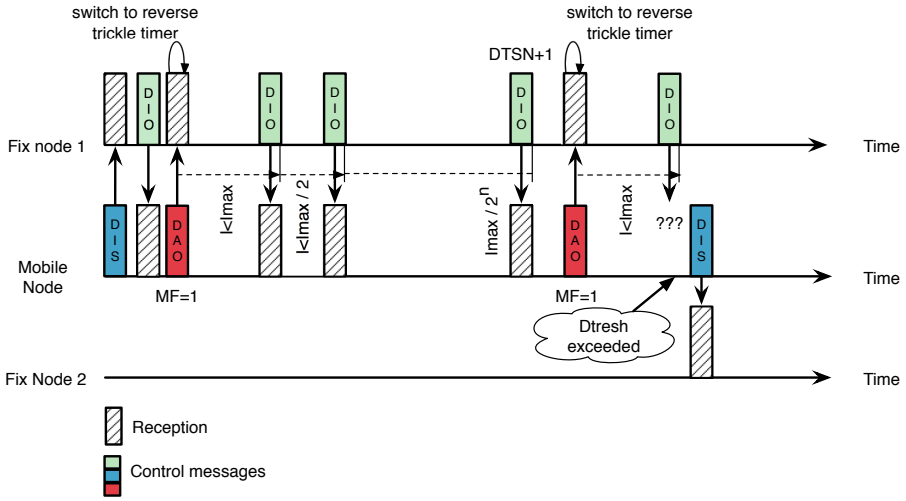


Figure 4.2: Proposed algorithm

When *Dthresh* is exceeded, the mobile node removes the parent and sets its rank to infinite. Then, it starts sending DIS messages in multicast to discover new parents, as the DIS message will reset the standard trickle timer in RPL and will determine nodes to send new DIO messages. The mobile node resets its rank to infinite, as this will give it the possibility to choose a new parent, regardless of parent's rank. After the mobile node receives new DIO messages, it will choose a parent to which it will also send DAO messages with MF flag set to 1. The messages exchanged between a parent and a mobile node, to manage the mobility of the mobile node are illustrated in Figure 4.2.

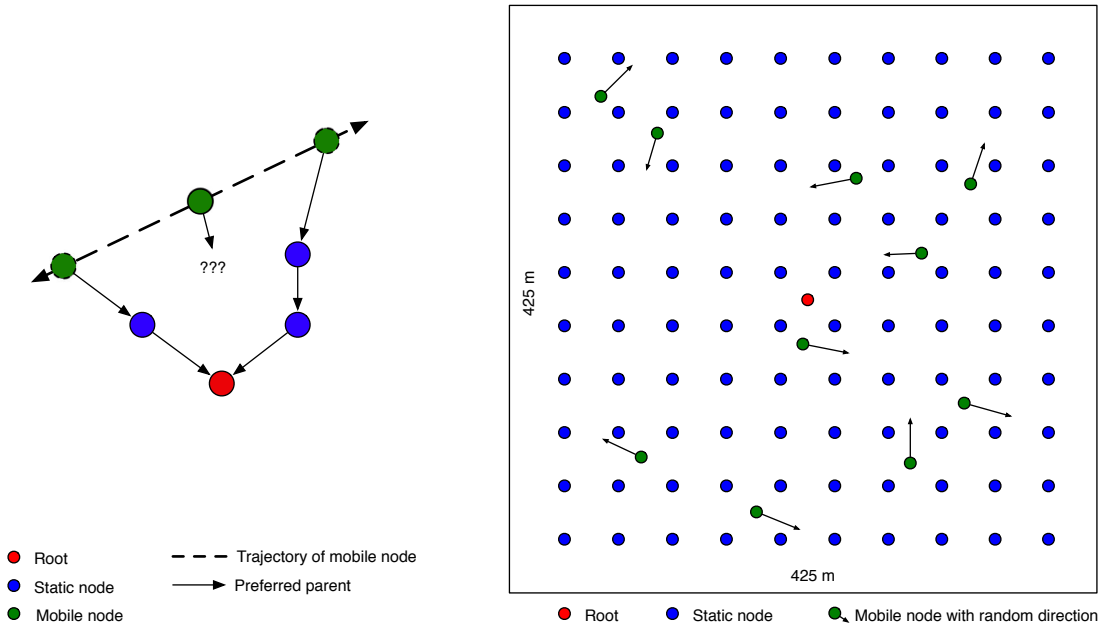
In the next section, we will evaluate our proposal with two state of the art solutions found in the literature. First, we will present the simulation setup, then we will analyze parameters such as disconnection time or control message overhead.

4.3 Simulation Setup and Results

4.3.1 Simulation Scenario

In order to evaluate RPL's performance when operated with mobile nodes, we used the WSNet software [Wsn]. WSNet is a discrete event simulator dedicated to the study of wireless sensor networks. WSNet already provides a basic RPL module that we extended to operate as presented in the previous section. Furthermore, we implemented two solutions from the literature referred to as PeriodicDIO [Lee+12] and DynamicDIS [Kor+12], which were detailed in Section 3.5.3 of the previous chapter.

All simulation parameters are presented in Table 5.1. We deployed two topologies. The first topology is for an academic scenario in which mobile nodes can only choose one parent. This topology is presented in Figure 4.3a. The second topology aims to provide a realistic environment in which the DODAG root is located in the middle of a 425x425m simulation area, 100 static nodes are positioned as a regular square grid, in a way that the whole simulation area is covered at the radio level and finally, 10 mobile nodes are initially distributed in a random way over the simulation grid and move randomly inside the area covered by the static nodes (as seen in Figure 5.3b). In order to better evaluate the impact of the proposed solution on the network,



(a) Academic (b) Grid

Figure 4.3: Analyzed topology

we chose to "freeze" the DODAG built between static nodes (i.e. static nodes will not change anymore their parent once the mobile nodes start moving in the simulation area). In the following, we will analyze the performance obtained in simulation with each solution.

4.3.2 Simulation Results

The results presented in this section were obtained after running 20 simulations of each scenario in every configuration. The presented results are the average of overall data collected from the set of simulations. The 95% confidence interval indicates the reliability of our measurements. We analyzed three main parameters: overall number of DIO messages sent in the network, mobile node disconnection time from the preferred parent and packet delivery ratio.

In the academic scenario, the mobile node will change only once the preferred parent. The mobile node moves on a path described in Figure 4.3a. We analyzed this simple scenario to better understand how a handover is performed. It is a good measure of how handovers will affect both the mobile node, on its path in the network, and the static nodes that it chooses as parents.

It can be seen in Figure 4.4 that, if only the parent to which the mobile node connects sends DIO messages, the overall number of control packets is reduced. In our solution and in DynamicDIS, DIO messages are sent only locally. The two solutions send less control packets than PeriodicDIO, in which the entire network is flooded with DIO messages. Sending DIO messages only when needed reduces the disconnection time of the mobile node from the DODAG. Parent change time, presented in Figure 4.5, shows that relying only on the RPL mechanisms to change the parent (as DynamicDIS and PeriodicDIO do), can take a long time. ETX needs traffic to be present in the network to be effective, and so the mobile node needs to receive many DIO messages before changing its parent. In contrast, our solution changes the parent of the mobile node independently of metric. When *Dthresh* is exceeded, the mobile node will remove all parents. Therefore, as soon as a new DIO message is received, the mobile node will quickly choose a new parent, regardless of the advertised rank. Waiting for only *Dthresh* for a new DIO message before considering that we are disconnected, limits the disconnection time to *Dthresh*. If *Dthresh* is exceeded, the mobile node sends a new DIS message in multicast to trigger new

Simulation parameter	Value
Academic scenario **Realistic scenario**	1 root, 3 static nodes, 1 mobile node 1 root, 100 static nodes, 10 mobile nodes
Data collection scheme	Time driven 1 packet/10s static nodes → root 1 packet/s mobile node(s) → root
Data payload size	56 bytes (in a 127 bytes IEEE 802.15.4 packet)
Mobility model	Billiard, 1m/s, linear trajectory (academic scenario) and random trajectory (realistic scenario)
Routing model	RPL in non-storing mode DODAG build using ETX metric
RPL default values	DIO - given by trickle timer algorithm [Dev+11] DIS - 60s if no parent, until attached to DODAG DAO - 60s from every node, or when needed
Analyzed schemes Reverse trickle algorithm	 $I_{min1} = 2^{10}$ms and $I_{max1} = 2^{24}$ms $I_{min2} = 2^{11}$ms and $I_{max2} = 2^{25}$ms $Dthresh$ is set to 1, 3 and max
PeriodicDIO [Lee+12]	1 DIO/2s and 1 DIO/10s
DynamicDIS [Kor+12]	DownDIS = 1, UpDIS = 5, I_{min} = 3s, I_{max} = 60s, I_{init} = 3s or 30s
MAC model	Unslotted CSMA 802.15.4 (802.15.4-2006)
Radio model	Half-duplex, Sensibility level: -92dBm, Channel 0, 250 kB/s bandwidth, 50m disk range
Antenna model	Omnidirectional, modulation OQPSK
Simulation setup Academic scenario Realistic scenario	20 simulations/configuration 10 configurations (including PeriodicDIO and DynamicDIS schemes) 1 handover/simulation 1 hour/simulation

Table 4.1: Simulation parameters

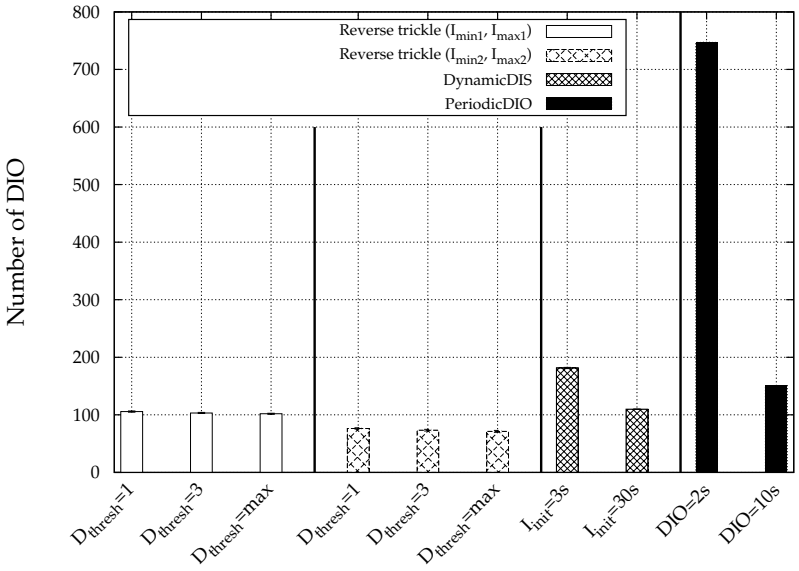


Figure 4.4: Number of DIO sent - academic scenario

DIO messages from neighboring nodes. Lower disconnection time translated to higher Packet Delivery Ratio (PDR) values.

Having a short disconnection time, as seen in Figure 4.5, enables our solution to have the highest PDR of all analyzed solutions when using I_{min1} and I_{max1}. This enables a quick parent change and low disconnection time from the DODAG. Because of the way ETX works, PeriodicDIO and DynamicDIS lose many packets, as the time to change the parent is longer. Several data packets need to be exchanged by the static nodes in order to have different ETX advertised values in sent DIO messages. Our solution sends less control packets, but the mobile node remains connected longer to the graph and has higher PDR for intervals $[I_{min1} - I_{max1}]$.

Moving on to the realistic scenario, before analyzing the results, we present the assumptions made. The DODAG that RPL build needs a long time to stabilize [Iov+13]. Therefore, we started analyzing our solution only after 30 min from the start of the simulation, when the DODAG will be in a stable state. When the mobile node enters the network, all possible parents will have already a preferred parent (and a path to the root). From now on, static nodes will not change anymore the preferred parent, enabling us to focus only on interactions in the network introduced by the mobile node.

Control packets sent by all the nodes in the network, shown in Figure 4.6, give now a clearer view of how localized our solution is. The mobile node will determine an increase of the control traffic only on nodes that it connects to, not in the entire network. Using DynamicDIS, mobile nodes send DIS in multicast and reset the trickle timer for neighboring nodes, increasing unnecessary the overall control traffic. PeriodicDIO, floods the entire network with control packets, even from nodes that will not be parents for the mobile node.

As many data packets are exchanged between the static nodes and the root, ETX will be able to easier distinguish between parents when a choice must be made. Thus, DynamicDIS and PeriodicDIO will allow the mobile node to change faster the parent, as seen in Figure 4.7. Still, as our solution does not depend on the ETX metric to change the parent, the disconnection time in this scenario is comparable to the one obtained in the academic scenario. We show again that relying on the RPL built-in mechanism to change the parent can take longer than our solution. In the realistic scenario, we managed to have better PDR values than the other two solutions. As just a few packets manage to arrive at the root from the mobile nodes, we analyzed why packets are lost. We identified three main causes:

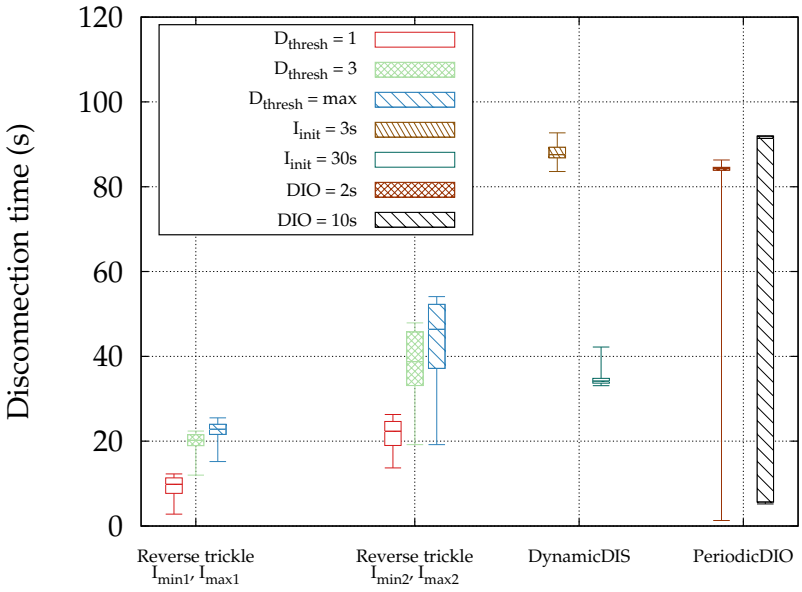


Figure 4.5: Delay mobile node handover - academic scenario

- mobile node has not yet chosen a new parent and is disconnected from the DODAG
- packet loss on the link between the mobile node and the preferred parent
- packet loss on the path from the preferred parent to the root.

Our solution, for $I_{min1} - I_{max1}$, loses 3.5% of packets when the mobile node did not choose yet a new preferred parent, between 7.5-15% of packets are lost on the link between the mobile nodes and the parent and 73-81% on the path from the preferred parent to the root. When we change the values to $I_{min2} - I_{max2}$, loss due to mobile node's lack of parent remains the same, but between the parent and the mobile nodes we lose 22-32% of packets and the rest (53-63%) on the path from the preferred parent to the root. DynamicDIS and PeriodicDIO always provide a parent for the mobile nodes. Nevertheless, 48-52% of packets for PeriodicDIO and 55% for DynamicDIS are lost as the mobile node believes it is still connected to the parent, but has moved already outside the range of the later. The rest (41-44% for PeriodicDIO and 36% for DynamicDIS) are lost on the path from the preferred parent to the root.

From the two analyzed scenarios, we have seen that our second contribution in the thesis has better performances than the two other state of the art solutions. Our solution enables us to limit the disconnection time, lower the number of DIO messages sent, manages to achieve if not higher, than a comparable PDR, and loses less packets on the link between the mobile node and the parent. Choosing $I_{min}1 - I_{max}1$ and $Dthresh = 3$ gives optimal performance in the network, if we consider all analyzed parameters. With this values control traffic is limited, PDR values remains high and loses are mitigated.

4.4 Conclusion

In this chapter, we have analyzed the performance of our second contribution that brings enhancements of RPL operations for mobility management. We have identified in Section 3.5.1 of the previous chapter, the problems that may raise by using RPL built-in mechanism to manage the parent set applied in a mobility situation. The presented enhancements do not change the topology of the network. Interactions remain localized, only between the mobile node and the chosen preferred parent. Once a node detects that in its sub-DODAG there is a mobile node,

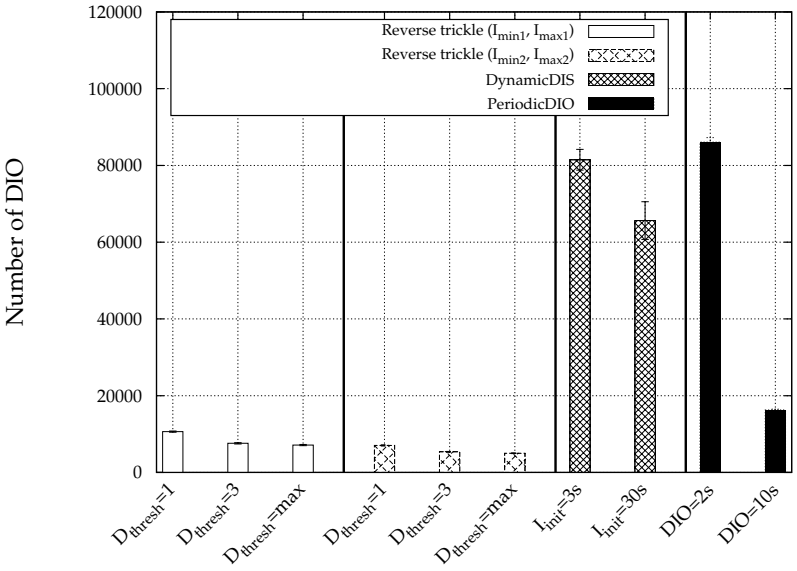


Figure 4.6: Number of DIO sent - realistic scenario

it will pause the standard trickle timer present in RPL and will switch to our proposed reverse trickle timer. This will enable a closer tracking of the mobile node. The mobile node itself has tools to determine when the preferred parent is unreachable, therefore if the defined threshold is exceeded, this will trigger a new parent search. In this way, the mobile node can maintain connectivity with the DODAG in an efficient way.

Simulations made using the popular ETX metric show that our second contribution allows mobile nodes to reduce the disconnection times experienced while moving from one parent to another. Furthermore, our solution outperforms the two other chosen solutions from the literature in terms of traffic overhead and disconnection times. This is due to the localized interactions between the mobile node and the preferred parent and to the reverse trickle timer algorithm. Packet delivery ratio is higher in the majority of the analyzed configurations. Packets are lost less on the link between the mobile node and the parent than in the other two proposals. In the light of all those results we can thus confirm our claim that the enhancements brought by our second contribution improve the efficiency of managing mobility in RPL.

Nevertheless, there still remain some problems to be addressed. The mobile node still needs to actively monitor the connection with the preferred parent, as it will synchronize itself with the received DIO messages. Having a mobility management solution just at the networking layer can lead to long disconnection of the mobile node at the MAC layer. We have seen in Chapter 2 solutions to manage mobility at the MAC layer, so leveraging information from such a MAC protocol in RPL protocol should bring improvements for communication. In the next chapter we will use the most efficient MAC layer identified in Chapter 2 that provides mobility management, X-Machiavel, and will leverage its actions in RPL protocol through an external unreachability detection mechanism.

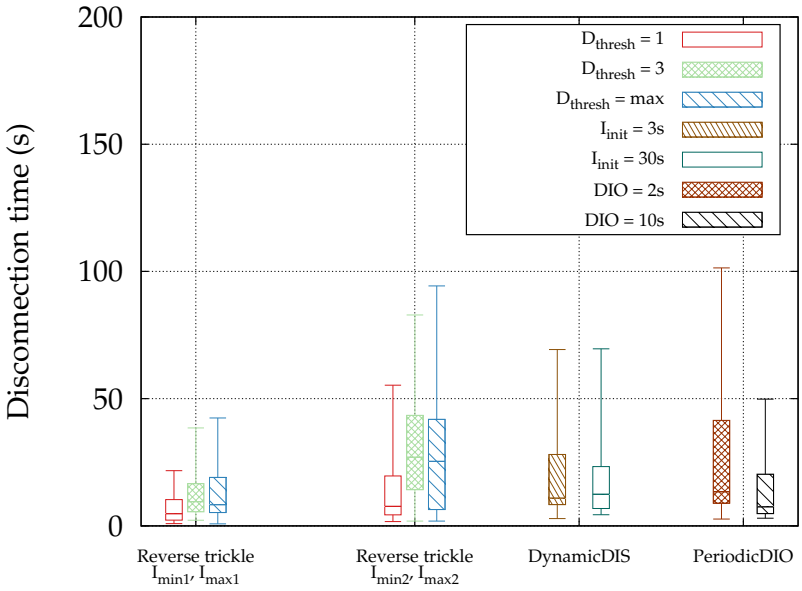


Figure 4.7: Delay mobile node handover - realistic scenario

		Academic scenario		**Realistic scenario**	
Mobility support solutions		Avg.	±	Avg.	±
Reverse trickle (I_{min1},I_{max1})	Dthresh=1	93.68	1.98	8.92	2.38
	Dthresh=3	89.08	1.88	9.22	2.79
	Dthresh=max	87.50	2.36	8.90	2.12
Reverse trickle (I_{min2},I_{max2})	Dthresh=1	86.46	2.50	9.05	2.58
	Dthresh=3	73.47	3.92	9.81	3.38
	Dthresh=max	71.94	3.90	8.75	2.66
PeriodicDIO	DIO/2s	71.62	5.70	8.12	1.60
	DIO/10s	71.94	3.90	8.75	2.66
DynamicDIS	I_{init} = 3s	64.95	0.37	8.33	1.70
	I_{init} = 30s	74.98	0.28	8.23	1.73

Table 4.2: Packet Delivery Ratio with 95% confidence intervals for academic and realistic scenarios

Chapter 5

Mobility Triggered RPL

In the previous chapter we have presented enhancements of RPL operations that provide mobility support. As these enhancements are present only in RPL and the solution provides mobility support only at networking layer. Nevertheless, the mobile node may become disconnected at the MAC layer, which will lower the performances of mobility management. In Chapter 2 we have made an overview of MAC protocols that support mobility management and concluded that X-Machiavel offers the best support for mobility at MAC layer. As we have seen, solutions for mobility management are available at both MAC and networking layer. We can bridge the gap between these two layers using hints from lower layers via Layer 2 (L2) triggers and leverage information from the MAC layer into the networking layer. In a general fashion, a cross-layer approach that is reactive to layer 2 events should be more efficient in providing unreachability information and offer mobility support. There are some MAC protocols, like IEEE 802.15.4 in beacon-enabled mode, that keep track of nodes associated to a PAN coordinator and detecting disconnection is already implemented in the protocol, but this is unavailable in most WSN MAC protocols. Other MAC protocols need to use L2 triggers, which allows events to be faster delivered to upper layer protocols.

In this chapter, we propose a new cross-layer protocol that manages network dynamics in WSN. Mobility-Triggered RPL (MT-RPL) is a specific implementation of L2 triggers linking RPL and X-Machiavel preamble sampling MAC protocol. We will start by describing how unreachability detection mechanisms can be integrated with RPL to achieve mobility management. Then, we will present our main contribution in this thesis, namely Mobility Triggered-RPL (MT-RPL). Performance evaluation of the proposed contribution is done through both simulation and experiments alongside NUD and BFD. The chapter will end with conclusions regarding the performance of the main contribution in the thesis.

Contribution

In this chapter we will present the following contributions to mobility management in RPL with external unreachability detection mechanisms:

1. A new cross-layer mechanism that leverages information from MAC layer (X-Machiavel MAC protocol) to the networking layer (RPL routing protocol), which constitutes our main contribution in the thesis.
2. A proposal on how NUD and BFD unreachability detection mechanisms should be used with RPL to enable mobility management.
3. A thorough analysis through both simulation and experiments of the envisaged solution (MT-RPL), along the other suggested unreachability detection mechanism (NUD and BFD).

5.1 Unreachability detection mechanisms coupled with RPL

We have seen how Neighbor Unreachability Detection, Bidirectional Forwarding Detection or hints from lower layers via Layer 2 (L2) triggers work in Section 3.5.2. Now, we will also propose how they would interact with RPL to enhance RPL mobility management capabilities.

– Neighbor Unreachability Detection maintains connectivity information between neighbors in a neighbor cache. When a path to a neighbor appears to be failing, NUD signals the need for a new next hop in RPL by deleting the neighbor cache entry. At RPL layer, this will allow the mobile node to remove the parent from the parent set. If the parent is the preferred parent, RPL will start searching for a new one, either in the parent set (if it is not empty) or through a local repair (as we have previously seen - the mobile node will delete all parents in the parent set, will set its rank to infinite and send multicast DIS messages in the neighborhood). The default timers of NUD can detect the unreachability of a neighbor in up to 38 sec. We doubt that such delay is short enough to allow RPL nodes to change parent seamlessly and without experiencing packet loss.

– Bidirectional Forwarding Detection monitors the number of control packets a system misses in a row. If more packets that the defined number are missed (specified as a function of time in which the system is still in UP state), BFD declares the route to the neighboring system DOWN (by default after a BFD packet is not received). At RPL layer, this will allow the mobile node to remove the respective parent. If the parent is the preferred parent, RPL will start searching for a new one, either in the parent set (if it is not empty) or through a local repair. RPL is paired with BFD in asynchronous mode. In this mode, BFD messages are sent periodically between systems, to detect any disconnections between the parent node and the mobile node. The default value between BFD packets is 20 sec. With BFD, greater care must be taken in choosing the periodicity of BFD packets, as both sides of the communication send them. Again, we doubt that the default delay in BFD has a value low enough that will allow a mobile node to make a seamless parent change (without loosing data packets) in RPL.

– Hints from lower layers via Layer 2 (L2) triggers signal events from layer 2 at upper layers. With RPL, Type 2 primitives are the most efficient way to notify RPL of events from layer 2. Once a trigger is registered, events are reported asynchronously each time they occur. In the next section, we will describe our main contribution, MT-RPL, that takes advantage

of triggers in an attempt to improve mobility management in RPL and achieve seamless parent change.

5.2 Mobility-Triggered RPL

Our main contribution in the thesis is built on top of X-Machiavel MAC protocol and is implemented on mobile nodes. X-Machiavel prioritizes the transmission from mobile nodes. To take advantage of this in MT-RPL, RPL registers a L2 trigger to be informed asynchronously every time the mechanism of X-Machiavel is triggered (e.g. channel stealing or using an opportunistic forwarder) on mobile nodes. In RPL protocol, the received triggers will lead to a change of parent that reflects how the transmission at the MAC layer has occurred (i.e. the node that receives the data from the mobile node at MAC layer will become the new preferred parent in RPL). To achieve this, MT-RPL includes the rank computed at RPL layer in the MAC header. Nodes can thus decide in a distributed way whenever it is worthwhile to act as an opportunistic forwarder (when the node is static) or to steal the medium from an ongoing communication (when the node is mobile). MT-RPL operations are presented in the following.

If the communication channel is free, a mobile node sends a P0 type preamble including its rank from the RPL layer. If the advertised destination in the preamble is in the neighborhood, X-Machiavel principles apply: the destination node sends a PK1 acknowledgement, claims the data from the mobile node and forwards it further to the root. This means that at RPL, the preferred parent of the mobile node has received the data packet, so no change will be made in the mobile node parent set.

When the advertised destination in the preamble is not in the neighborhood, another static node may receive the P0 preamble from the mobile node. The static node can decide to act as an opportunistic forwarder for the pending data packet. If the rank in the P0 preamble is greater than the rank of the opportunistic forwarder (i.e. the mobile node is located further in the graph than the opportunistic forwarder), the opportunistic forwarder can send back a PK0 acknowledgement (as seen on the left side of Figure 5.1). Upon reception of the PK0 acknowledgement, the mobile node changes the destination of the data packet to the address of the opportunistic forwarder and sends the data packet to it. If the mobile node receives an acknowledgement from the opportunistic forwarder for the data packet, a trigger will allow the mobile node to change the preferred parent in RPL (i.e. the node that acts as an opportunistic forwarder and acknowledged the data packet is set as the new preferred parent of the mobile node). The opportunistic forwarder will send the data packet from the mobile node using P2 preambles at the MAC layer, so that no other mobile nodes can steal the channel, following the DODAG until the root. Potential forwarders with a rank equal or greater than the one of the mobile node, simply discard the overheard preamble.

Transmitting data on an occupied channel requires the mobile node to seize the opportunity to transmit its data between two consecutive strobed preamble frames that are destined to another node. X-MAC principles require that the destination of preamble strobes send back an acknowledgement between two strobes to notify the sender to stop the preamble and proceed with the data. MT-RPL, as it uses X-Machiavel MAC protocol, allows mobile nodes to send their own data if an acknowledgement from the original destination is not received (as explained in Section 2.3.1). However, MT-RPL enables this behavior only if the rank of the sender of the preamble is lower than the rank of the mobile node (i.e. the mobile node's data will make forward progress toward the root of the DAG). As a result, channel stealing in MT-RPL operates as follows. First, a mobile node should overhear a P1 preamble destined to another node and announcing a RPL rank lower than its own RPL rank. Then, the mobile node changes the destination of its data packet to the address of the sender of the preamble and transmits the resulting packet before the next preamble strobe (as seen on the right side of Figure 5.1). The static forwarder, once it receives the data packet from the mobile node, will forward it towards the root, using P2 preambles at MAC layer. If the mobile node overhears a P2 preamble from the static node to which it has sent the data packet, a trigger will allow the change of the preferred

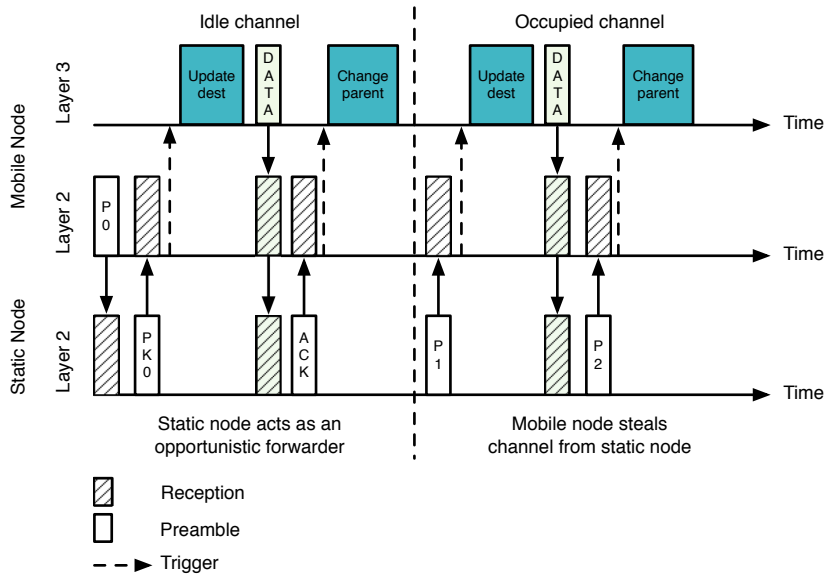


Figure 5.1: Preamble is acknowledged or overheard

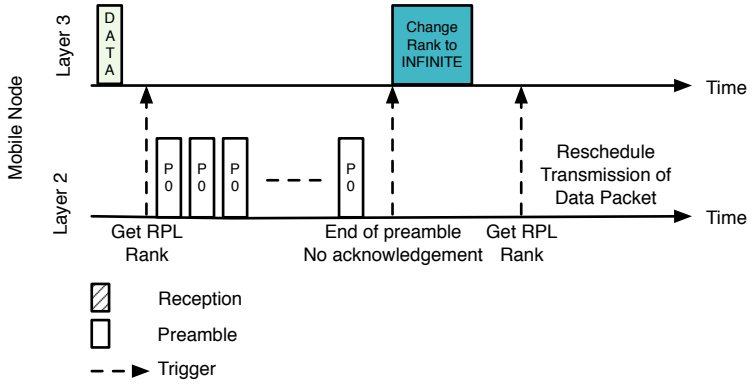


Figure 5.2: Preamble is not acknowledged

parent at RPL (i.e. the static node from which the mobile node stole the channel becomes its new preferred parent). After forwarding such packet, the static node that acted as a forwarder still needs to send its own data (as it has advertised a P1 preamble before announcing that it has data to send) and does that by using P2 preambles. Further along nodes operate as in X-MAC. Regardless of how the static nodes received data packets from mobile nodes, they will always forward it using P2 preambles until the final destination is reached.

If the preamble sent by a mobile node is not acknowledged (Figure 5.2), this means that the mobile node is in an area where all surrounding nodes have a rank higher than its own. A trigger will notify RPL that no parent is available, so the mobile node becomes disconnected from the DODAG. Generally, at the MAC layer, packets that are not successfully sent are retransmitted. The mobile node will advertise an infinite rank in the MAC header of the next preamble retransmission. Therefore, any neighbor that receives the preamble can now acknowledge it. The mobile node sends the data packet to the static node, and if an acknowledgement for the data is received, a trigger will notify RPL to change the mobile node's preferred parent to the static node that just acknowledged the data.

In the former paragraphs we described how MT-RPL leverages X-Machiavel actions at the networking layer in RPL protocol. The mobile node may send a data packet to a static node

that either acts as an opportunistic forwarder or has had his channel stolen by the mobile node. When the mobile node steals the channel, the static node may not be in the mobile node's parent set. Nevertheless, the mobile node has the MAC address of the static node that should become the preferred parent at RPL (as the static node acknowledged the mobile node preamble or the mobile node has overheard a preamble from the static node). The IPv6 address of the static node can be thus inferred from the MAC address, as we use 6LoWPAN. We can thus avoid sending a multicast DIS message in the neighborhood, that will reset trickle timers at any receiving node. Instead, we can send a unicast DIS to the static node. After the static node receives the unicast DIS message, it will reply with a unicast DIO message. The mobile node can add the static node as a preferred parent after processing the received unicast DIO message.

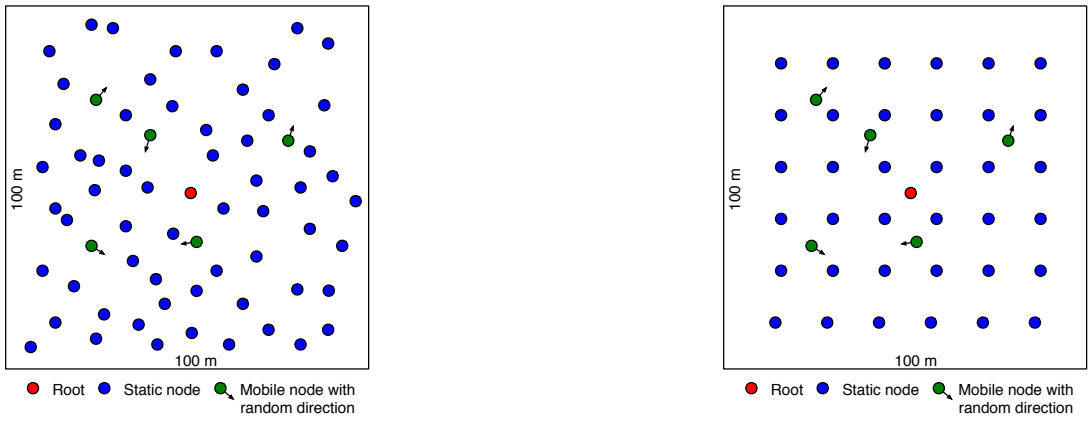
To summarize, MT-RPL manages the parent set with regard to the information received from the MAC layer through L2 triggers. When the mobile nodes benefits from an opportunistic forwarder (by receiving a PK0 acknowledgment) or steals the medium from another node (sending a data between two consecutive preamble strobes for an ongoing communication), if the transmission is successful, the MAC layer provides the rank and the address of the effective next hop in RPL. Upon reception, RPL sets this node as the new preferred parent, computes the related rank and proceed with RPL operations whenever necessary (send new DAO and/or DIO messages). As a result, MT-RPL should enable smooth preferred parent change during mobility management by enabling nodes to promptly react to network change without generating extra control traffic.

These final claims will be verified through an extensive simulation and experimentation campaign and will be compared to results obtained with NUD and BFD unreachability detection mechanisms in the same network conditions. We continue to present the simulation scenarios and obtained simulation results.

5.3 Simulation Setup and Results

5.3.1 Simulation Scenario

In order to evaluate the mitigation of network dynamics by RPL, we used the WSNet software [Wsn]. WSNet is a discrete event simulator dedicated to the study of wireless sensor networks. WSNet already provides a basic RPL module that we extended to operate as presented in both Section 5.1 and Section 5.2.



(a) Random (b) Grid

Figure 5.3: Analyzed topology

All simulation parameters are presented in Table 5.1. We deployed a random topology of 60 nodes on a 100x100 m area with the root in the middle (as seen in Figure 5.3a) and a

Simulation parameter	Value
Topology	
Random topology Grid topology	1 root, 60 static nodes, 5 mobile nodes 1 root, 36 static nodes, 5 mobile nodes
Data collection scheme	Time driven 1 packet/30 sec static nodes → root 1 packet/5s mobile nodes → root and root → mobile nodes
Data packet size	127 bytes
Mobility model	Billiard, 1m/s random trajectory
Routing model	RPL in non-storing mode DODAG build using MinHop metric
RPL default values	DIO - given by trickle timer algorithm [Dev+11] DIS - 2s if empty parent set, until attached to DODAG DAO - 60 sec from every node, or when needed
Values for parameters of unreachability detection mechanisms	
NUD (RFC 4861)	Maximum number of NS transmission - 3, Delay first probe - 5s, Reachable time - 30 sec, Retransmission time - 1s
BFD (RFC 5880)	Desired TX interval - 30 sec, Missed BFD packets that bring session DOWN - 1
MAC model	X-MAC (for standard RPL, NUD and BFD) and X-Machiavel (for MT-RPL) Maximum number of retransmissions - 4
Radio model	Half-duplex, Channel 0, Sensibility level: -92dBm, 15 kB/s bandwidth, 18m (60 feet) [Pol+05] unit disk range
Current consumption	TX: 31 mA, RX: 15.1 mA OFF: 400 nA (CC1100 chip)
Antenna model	Omnidirectional, modulation BPSK
Simulation setup	20 simulations/mechanism/topology, 4 mechanisms, 2 topologies, 1 hour/simulation

Table 5.1: Simulation parameters

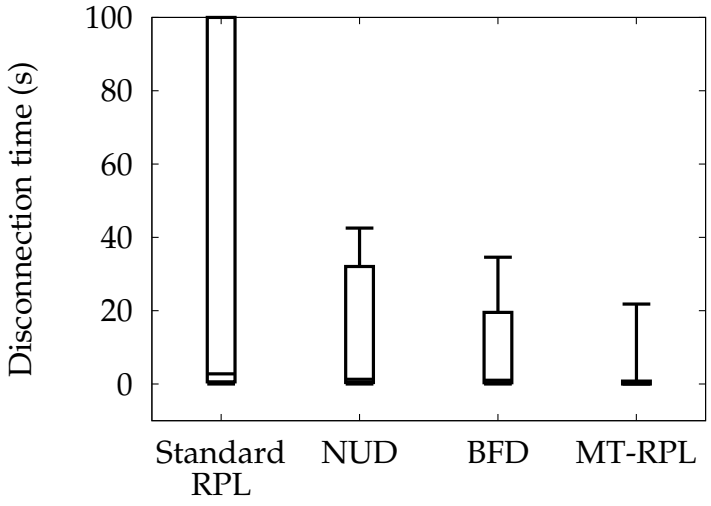
grid topology with 36 nodes and the root in the middle (as seen in Figure 5.3b). To generate dynamism, 5 mobile nodes are distributed and move following a simplified version of the random direction model, used also in [HP01]. Such nodes are pre-configured with the status of mobile node, as they have physical capabilities to move (e.g. the node is on a platform with wheels). The built-in mechanism in RPL that manages the parent set will be referred in this chapter as standard RPL. Standard RPL, NUD and BFD are coupled with X-MAC because X-Machiavel favors transmissions of data packets from mobile nodes, but the node which acknowledges the preamble, or from which the channel is stolen, may not be the parent at RPL layer and the packet even though it is sent, it will be dropped by the receiving node (as this node is not the intended destination at the networking layer). As it receives information from X-Machiavel, MT-RPL takes advantage of these changes and adjusts the transmission of data packets accordingly. Only links between the mobile node and its parent are monitored using BFD, NUD or MT-RPL. On the rest of the path toward the root, the packet is routed using standard RPL, as these links will not be affected by the movement of the mobile node in the network. With all methods, mobile nodes keep only the preferred parent in the parent list, which may change when a DIO message with a better rank is received or when the mobile node does a local repair. The path from the root to the mobile node is maintained up to date with DAO messages, thus changes in topology are reported to the root in a timely fashion. Packets are delivered following source routing set by the root. In the analyzed scenarios, both mobile and static nodes send control messages as needed in order to maintain connectivity to the DODAG. The DODAG that RPL builds needs a long time to stabilize [Iov+13]. Therefore, we started analyzing results only after 30 min from the start of the simulation, when the DODAG will be in a stable state and the mobile nodes start moving. After this time, the structure of the DODAG in the static part of the network will not be allowed to change anymore. This is done in order to be able to analyze only the changes induced by the mobile nodes in the network. At the end of the simulation, packets were not sent for 15 min, so that all queues of packets from all nodes could be emptied.

With the above-mentioned implemented setup, we continue in the next section to evaluate the performances of standard RPL, NUD, BFD and MT-RPL on managing the mobility of mobile nodes in the network.

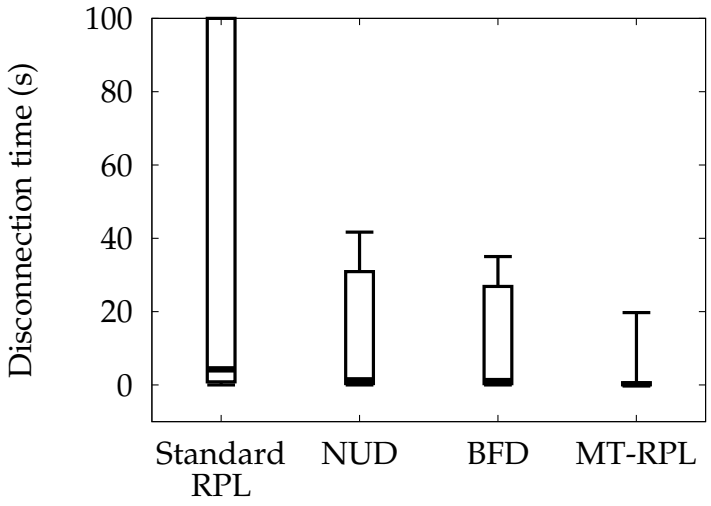
5.3.2 Simulation results

The results presented in this section were obtained after running 20 simulations of each scenario for each configuration for a total of 200 simulations. The presented results are the average of overall data collected from each set of simulations. The 95% confidence interval indicates the reliability of our measurements. We analyzed four parameters: disconnection time from the preferred parent, packet delivery ratio (PDR), overall number of control messages sent in the network and energy consumption.

Figure 5.9 illustrates the disconnection time for each scenario, i.e. the time between a mobile node going out of the radio range of its preferred parent and enforcing a new preferred parent at the RPL layer. As we can see, standard RPL shows the longest disconnection time (up to 700 sec. in the worst cases), as changing the preferred parent at the mobile node is done only by receiving a new DIO message with a better rank. Therefore, it is likely that a mobile node remains disconnected for a long period because all received DIO messages advertise a higher (worse) rank. An unreachability detection mechanism is therefore mandatory in order to avoid such situation that could lead to severe underachievement. By contrast, the disconnection time is drastically reduced when using RPL coupled with NUD, BFD or MT-RPL. With NUD and BFD, when a mobile node does not receive reachability confirmation from its preferred parent, RPL removes the preferred parent, resets the rank to infinite and starts sending DIS messages in multicast (i.e. the mobile node makes a local repair). BFD lowers the maximum disconnection time because it reacts quicker than NUD thanks to its slightly lower reachable time (30 sec versus 38 sec for NUD). Variations occur, as mobile nodes need sometimes to send several DIS messages before they can reconnect to the DODAG. Finally, MT-RPL presents the lowest disconnection times. Thanks to the interaction between the layers 2 and 3, a mobile node always regains



(a) Grid topology



(b) Random topology

Figure 5.4: Average disconnection time from parent

connectivity when an opportunistic node acknowledges its preamble and successfully receives the data packet. In addition, a mobile node regains connectivity whenever it successfully steals the medium from a neighbor node with a better rank. In those situations, the disconnection time is bound to the sending frequency of data packets and the number of preamble strobes sent before stealing the medium or opportunistic node acknowledgment. This explains the low disconnection time observed for MT-RPL in Figure 5.9. However, a mobile node may be in a situation in which it cannot steal the medium or opportunistic node cannot acknowledge its preamble strobes. Such situations occur when the mobile node moves in an area where the rank of all neighbors is lower (worse) than the rank of the mobile node. Nevertheless, MT-RPL allows a mobile node to reset its rank to infinite and remove its preferred parent after sending a whole preamble without receiving an acknowledgment, either from its preferred parent or from an opportunistic forwarder (as in Figure 5.2). As a result, in an unfavorable environment, the disconnection time is only increased with the transmission duration of a whole MAC preamble.

Grid topology			**Standard RPL**	**NUD**	**BFD**	**MT-RPL**
Mobile node to root	Packet delivery ratio	Avg. (%)	8.42	10.06	18.02	62.08
		± (%)	2.42	6.64	4.47	13.99
	Data packets sent	Avg.	666	184.61	501.84	410.15
		±	168.87	68.75	126.01	129.71
Root to mobile nodes	Packet delivery ratio	Avg. (%)	14.58	8.21	13.96	23.21
		± (%)	10.44	7.43	7.97	7.56
	Data packets sent	Avg.	23.95	22.46	47.42	64.60
		±	11.80	16.89	17.58	20.11
Random topology						
Mobile node to root	Packet delivery ratio	Avg. (%)	9.32	12.99	18.93	66.56
		± (%)	1.40	4.00	2.94	4.69
	Data packets sent	Avg.	895.36	210.87	482.78	757.17
		±	23.76	67.72	43.74	46.67
Root to mobile nodes	Packet delivery ratio	Avg. (%)	33.59	37.01	36.53	36.14
		± (%)	15.34	22.20	14.83	12.03
	Data packets sent	Avg.	34.00	22.25	40.57	81.17
		±	13.89	9.75	13.12	18.12

Table 5.2: Number of sent data packets and PDR with 95% confidence intervals

Lowering the disconnection time should increase the packet delivery ratio (PDR) on the paths from mobile nodes to the root and from root to mobile nodes. Note that we implemented the solutions so that mobile nodes only try to send data packets if a preferred parent is set. As a result, all solutions do not necessarily send the same number of data packets. Table 5.2 presents the PDR together with the number of data packet sent by each solution in the both scenarios. Standard RPL, as it cannot ensure continuous connectivity of mobile nodes to their parents, has the lowest PDR from mobile nodes to the root. In addition, with standard RPL, the mobile node sent the largest number of data packets because it has no means to remove an out-of-range preferred parent. Therefore, the mobile nodes keep trying to send data packets while their preferred parents are no longer reachable, increasing the packet loss together with the medium contention due to retransmissions. Results for the path from the root to mobile nodes are not meaningful because only few packets are actually sent. Most of the time, the root has no valid route to mobile nodes (DAO messages cannot be sent from mobile nodes when they are disconnected) and therefore buffers the packets. When an unreachability mechanism is present at the mobile nodes, PDR values improve. Thanks to BFD or NUD, mobile nodes change their

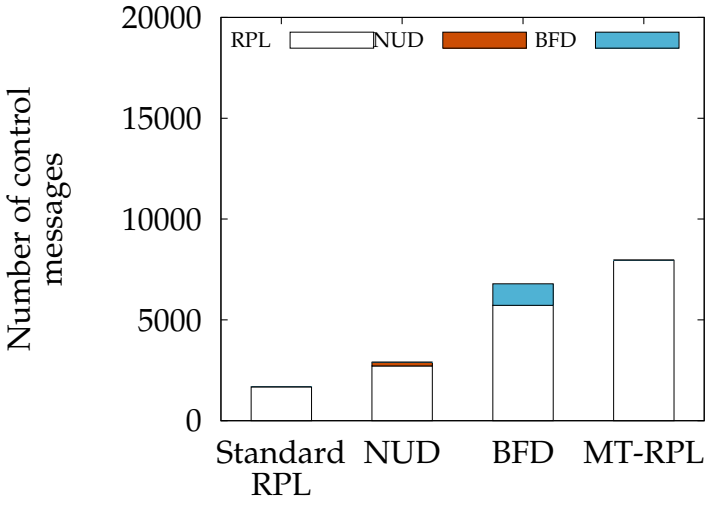
preferred parents more often, resulting in longer connections to the DODAG. This allows mobile nodes to send more data packets that successfully arrive at the root. However, values of PDR are still low, as the disconnection from the preferred parent may be reported after a long period of time (up to 30 sec for BFD and 38 sec for NUD). During this time, preferred parents are still considered as reachable, but all transmitted data packets are lost.

By contrast, lower disconnection times for MT-RPL seen in Figure 5.9 are translated into the highest PDR on paths between the mobile nodes and the root and on paths between the root and the mobile nodes. Channel stealing and opportunistic forwarding allow mobile nodes to connect to a parent with a better rank whenever possible. Such reconnections occur without triggering a local repair, reducing the disconnection time together with the signaling overhead. Neighbor nodes can keep thus a low transmission rate of DIO messages (they do not reset the trickle timer). However, data packets are still lost with MT-RPL as congestion can form on the path towards the root. The same observation is valid for the path from the root to the mobile nodes.

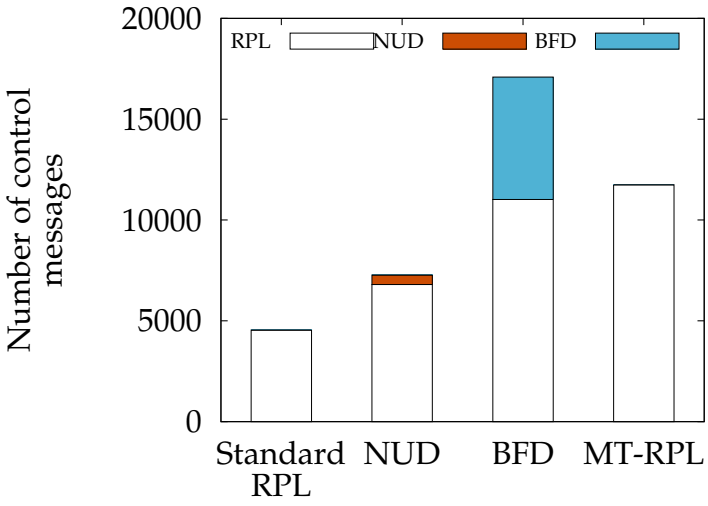
Figure 5.11 presents the signaling overhead of each solution. The low number of control packets sent in standard RPL further supports the assumption that the mobility of a node is rarely reported with this solution. Furthermore, discovering and attaching to a new parent is done only using the built-in mechanisms in RPL to change the preferred parent. Control messages are rarely sent, as the topology is in a stable state when the mobile nodes start moving. Adding unreachability detection mechanisms increases the signaling overhead in the network. Although BFD shows lower disconnection time and higher PDR than NUD, such results come at the expense of higher signaling overhead. BFD maintains sessions both ways between the mobile nodes and their parents by exchanging UDP packets every 30 sec (each node manages its own timer). This explains the increased number of BFD control messages in both topologies. NUD on the other hand, relays more on messages sent by the mobile node, which has to check periodically (every 38 sec) the connectivity to its parent. Furthermore, both NUD and BFD trigger a local repair when the unreachability of the preferred parent is confirmed. Such procedure resets the trickle timer of all neighbor nodes. After a local repair, DIO messages are therefore sent at a high rate, increasing the signaling overhead reported at the RPL layer. By contrast, MT-RPL does not introduce new control messages. In addition, parent change is achieved without triggering local repair, thus reducing the overall signaling overhead. However, MT-RPL increases the number of reconnections, and therefore makes the use of a large number of DAO messages to report each parent change.

Because energy consumption is one of the crucial points in WSN, we also evaluated the energy consumption of each node in the network. Results are reported in Figure 5.6. The Y-axis represents the energy needed to send 100 data packets in order to have a uniform representation for all methods. As a general remark, mobile nodes consume more energy than fixed nodes because they send 1 data packet every 5s whereas fixed nodes only send 1 data packet every 30 sec. With standard RPL, nodes try to send packets even if the parent is not in the neighborhood. If the preamble is not acknowledged and the retransmission number is reached, the data packet is dropped without being sent on the medium from the mobile node. This is why even with a large number of data packets sent by standard RPL energy consumption remains low, as only a few packets manage to actually be sent between nodes. Once mobile nodes are longer connected to their parent, the energy consumption for them and the root rises. NUD and BFD send additional control messages in the network. Given the low number of data packets sent by mobile nodes using NUD, the energy consumption needed to send 100 data packets is the highest of all. BFD, although it sends control packets both ways between the mobile node and parent, has lower energy consumption when we take into account the energy consumed for 100 data packets. Using only RPL control messages sent when changes occur in the network and are signaled by X-Machiavel, MT-RPL achieves the lowest energy consumption of all unreachability detection mechanisms.

These final remarks conclude the simulation part for standard RPL, NUD, BFD and MT-RPL. We have proven in simulation that our main contribution in this thesis has potential to lower mobile node disconnection time and bound it close to the data transmission rate. This will increase the PDR, as mobile nodes will find a new parent to send their data to easier and faster.

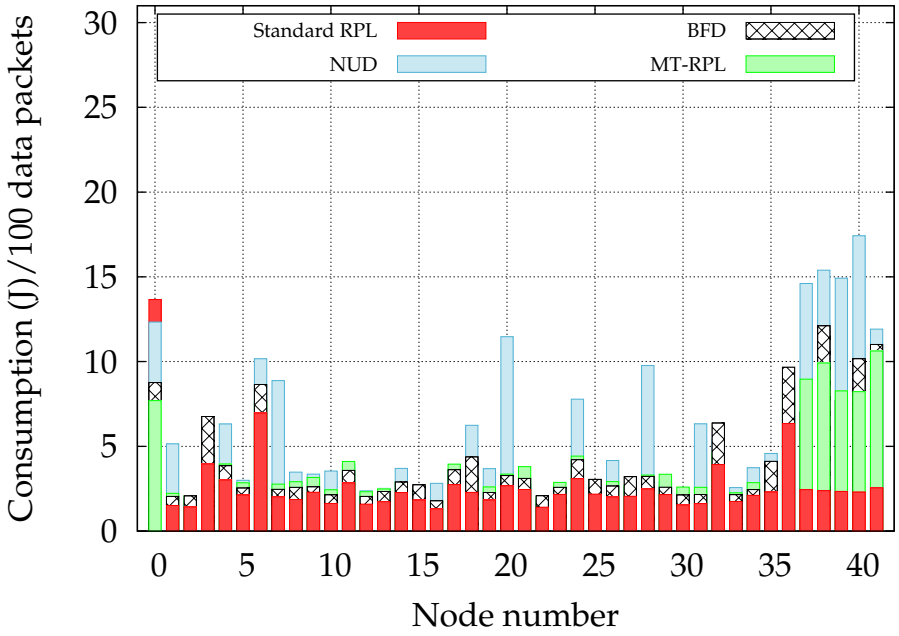


(a) Grid topology

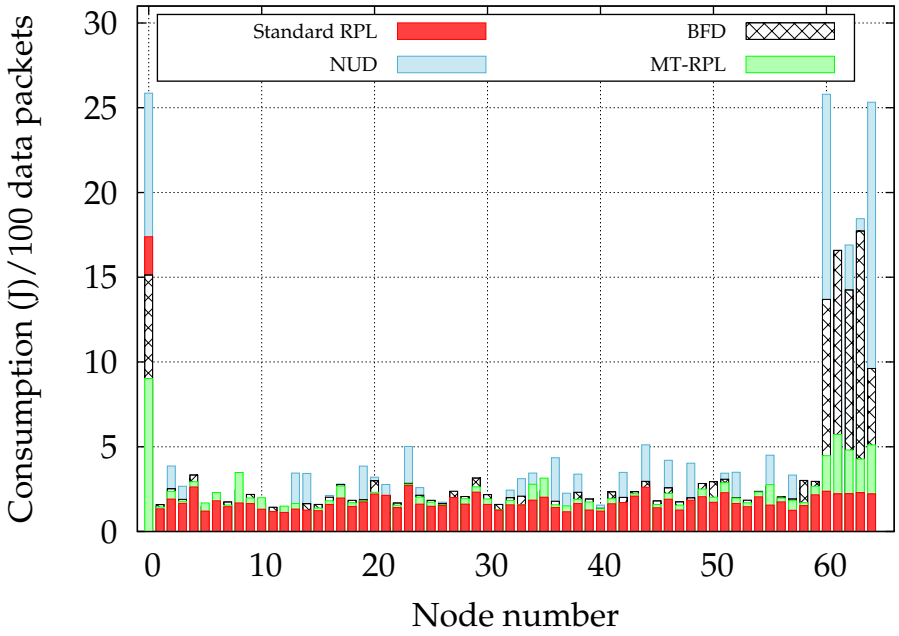


(b) Random topology

Figure 5.5: Average number of control messages sent



(a) Grid topology



(b) Random topology

Figure 5.6: Average energy consumption of nodes
Node 0 - root. Last 5 nodes - mobile nodes. Intermediate nodes - static nodes

Encouraged by these results, we continue on the FIT IoT-LAB [Fit] experimental platform and test the validity of simulation results also on a real implementation.

5.4 Experimental Setup and Results

5.4.1 Experimental Setup

The evaluation of the proposed solution, as well as those of the other unreachability detection mechanisms suggested by RPL is done on the FIT IoT-LAB [Fit] experimental platform. IoT-LAB is a large scale infrastructure facility aimed at enabling testing of innovative communication solutions designed for wireless sensor networks. The sensors deployed across 6 different sites in France offer researchers networks with different topologies. A variety of sensors are available, both in terms of processor architecture (MSP430 [Msp], ST2M32 [St2] and ARM Cortex A8 [A8]) as well as in terms of wireless chips (860 Mhz and 2.4 Ghz IEEE 802.15.4 PHY).

After implementing in Contiki OS all mechanisms presented in both Section 5.1 and Section 5.2, the deployment on the platform was done on Cortex M3 nodes (ST2M32 processor and 2.4 GHz IEEE 802.15.4 PHY). These nodes are the only ones available both in the static and the mobile part of the platform inside the Strasbourg and Grenoble sites.

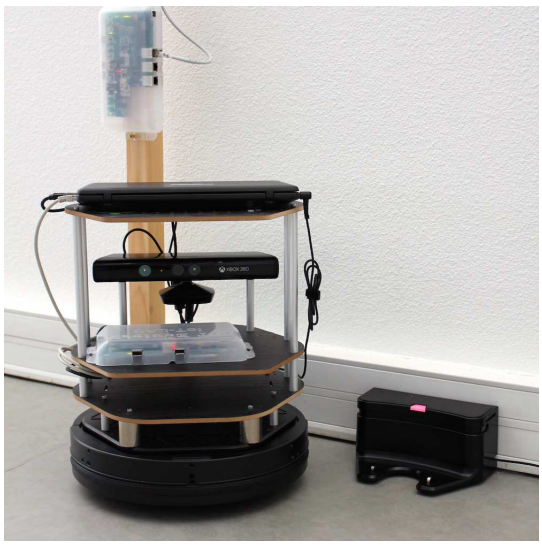


Figure 5.7: Turtlebot 2 robot

Mobility is central to our proposal and the IoT-LAB provides Cortex M3 nodes that are mobile, thanks to Turtlebot2 robots (seen in Figure 5.7), which move inside the testbed. The movement of the robot inside the testbed can be viewed as a Random Waypoint Model [JM96] with the following constraints. The speed of the robots varies. For instance, the robot will slow down before an obstacle so that the direction towards a waypoint can be adjusted. It also needs time to accelerate when departing from a waypoint and decelerate before it arrives to the next waypoint. The waypoints are considered reached when the robot arrives within a predefined range to the exact position of the waypoint. Once a waypoint is considered reached, the robot will stop and orient itself towards the next waypoint. The robot uses two types of correction when it comes to navigation: short range corrections, used to avoid any obstacles in the immediate vicinity, and long range corrections, that enable the robot to stay on track to the next set waypoint. We can see that by combining these two navigation methods it is possible that the path between waypoints is not always a straight line, as in the Random Waypoint Model.

Due to current limitations in the IoT-LAB testbed, the waypoints need to be defined before the experiment is launched. In return, the robot will loop between the defined waypoints during

the period of experimentation. This will allow a greater reproducibility of results over several experiments.

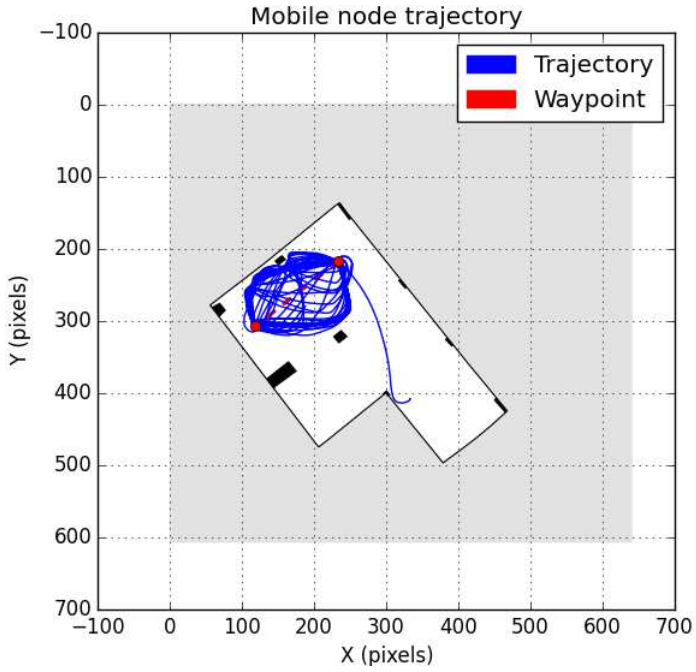
Experimentation sites have different characteristics. In the Strasbourg site, the static nodes are deployed in a form of a grid, with 2 m spacing between each other on all directions (X, Y and Z) and a line of sight (LOS) path between most nodes. Of the two layers of sensors that are available, we have chosen one layer only, forming a 5 x 5 node grid, which act as forwarders between the mobile node and the root and have LOS paths. The mobile node cannot choose the root as parent; this limitation was imposed by the physical dimensions of the site and the minimum transmitting power, or else the mobile node will always choose the root as parent and will never be disconnected, which will defeat the purpose of our contribution. The robot moves on the floor of the room, between 2 waypoints, defined before the start of the experiment (see Figure 5.8a for more details on the path of the mobile node). In the Grenoble site, the static nodes are deployed in corridors, beneath the walkway. The robot covers only a portion of the corridor, so it was natural to choose only the static nodes that are beneath the robot, as it travels between the defined waypoints (see Figure 5.8b for more details). We took advantage of the topology and moved the root of the DODAG outside the reach of the mobile node. In order to achieve this, we extended the static network on an adjacent corridor to the one where the robot moves, ensuring connectivity between all static nodes, as well as the mobile node. In both sites, we implemented a mechanism that allows us to force handovers. If the mobile node chooses a parent with low rank, it may stay attached during the whole experiment time. To avoid this, we implemented a timer that allows the static nodes to stop replying to requests from the mobile node, thus forcing the mobile node to change parent. When the timer is triggered on a static node, a multicast data message advertising the stopping of service for a mobile node is sent. This message, if received, is used by the mobile node to compute the time needed to detect the disconnection and reconnect to a new parent (it will not trigger any action to reconnect to the DODAG). Static nodes have two timers: one that sets the time the static node stops serving the mobile node (with values between 1 and 4 minutes) and another one that sets the time the static node will serve the mobile node (with values between 3 and 5 minutes). We have seen that these values always ensure that there is a parent available for the mobile node, while eliminating parent hopping (switching repeatedly between preferred parents, as they have similar ranks for the mobile node).

A more detailed view of the different parameters of experimentation is given in Table 5.3.

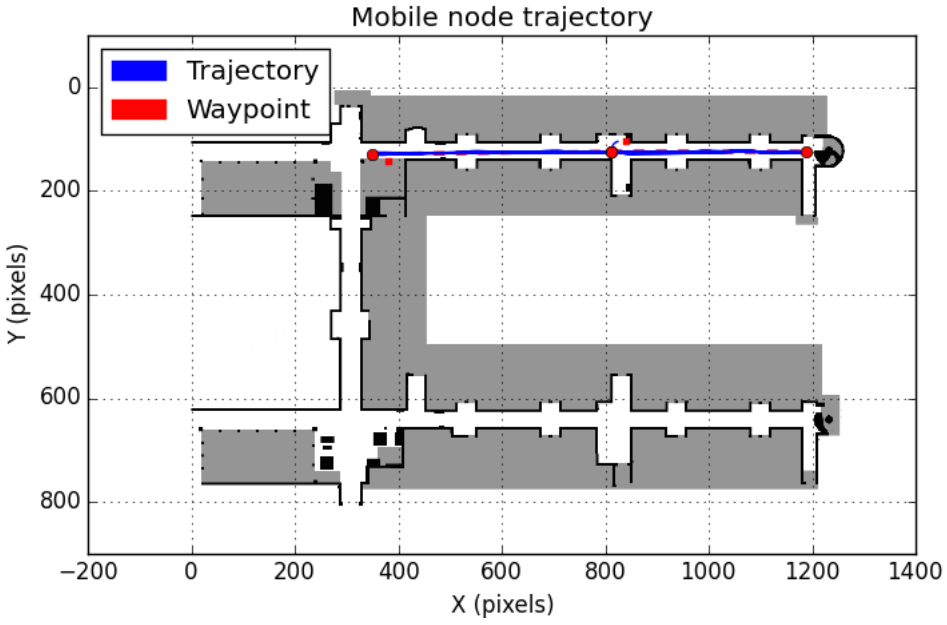
NUD and BFD are analyzed using X-MAC protocol so that only the destination node can acknowledge the data packet. We choose X-MAC as the same remarks made in the simulation setup in Section 5.3.1 are valid also in experimentation. On the other hand, MT-RPL uses X-Machiavel protocol, where any node with a better rank than the parent of the node can act as an opportunistic forwarder, acknowledging the data packet. In addition, the mobile node can steal the channel from a static node if it has a data packet to transmit. After the data packet is successfully forwarded through a new parent, changes are reflected at RPL layer. All mechanisms are used only between the mobile node and its respective parent; from the parent until the root standard RPL is used, as these links are not subject to change due to node mobility. The path from the DAG root to mobile node is kept up to date with DAO messages. Each change in topology is reported to the root. Nodes that receive DAO messages will also update their information about nodes in the sub-DODAG. With RPL in storing mode, nodes will take the routing decisions based on local information received in previous DAO messages. Mobile nodes will start communicating with the DODAG after 5 minutes from the start of the experiments. This period, considering the number of nodes in the topology, ensures a stable DODAG with few changes in the static part of the network. Static nodes choose a parent and set it's reachability to infinite, so only a DIO with a better rank can change their option.

5.4.2 Experimental results

For each mechanism presented above we made 10 experiments at each site (Strasbourg and Grenoble), leading to 60 experiments of an hour each. With a 95% confidence interval, our



(a) Strasbourg site



(b) Grenoble site

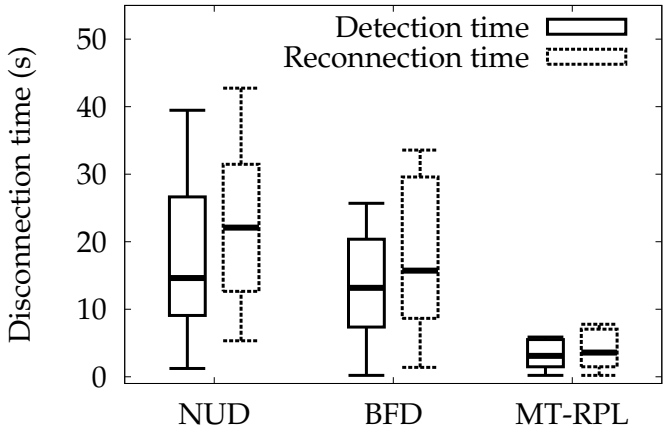
Figure 5.8: Trajectory of mobile node

Experiments parameter	Value
Topology	
Strasbourg site Grenoble site	1 root, 5 x 5 grid of static nodes, 1 mobile node 1 root, 10 static nodes beneath the walkway, 1 mobile node
Data collection scheme	Time driven 1 packet/30 sec static nodes → root 1 packet/5s mobile nodes → root and root → mobile nodes
Data packet size	127 bytes
Mobility model	Modified Random Waypoint, speed up to 0.8 m/s
Routing model	RPL in storing mode using ETX
RPL default values	DIO - given by trickle timer algorithm [Dev+11]; min. 4s, max. 8 doublings DIS - 60 sec or after each data packet if empty parent set, until attached to DODAG DAO - after parent attachment/change or when a DAO from a child node is received
Values for parameters of unreachability detection mechanisms	
NUD (RFC 4861)	Maximum number of NS transmission - 3, Delay first probe - 5s, Reachable time - 30 sec, Retransmission time - 1s
BFD (RFC 5880)	Desired TX interval - 30 sec, Missed BFD packets that bring session DOWN - 1
MAC model	X-MAC (for NUD and BFD) and X-Machiavel (for MT-RPL) Maximum number of retransmissions - 4 Duty cycle - 1/64 sec
Microcontroller unit	ARM Cortex M3, 32-bits, 72 Mhz, 64kB RAM ST2M32F103REY
Radio communication	IEEE 802.15.4 AT86RF231 transciever 250 kB/s bandwidth, TX power: -17 dBm, Sensitivity -101 dBm
Antenna model	Omnidirectional, modulation O-QPSK
Simulation setup	10 experiments/mechanism/site, 3 mechanisms, 2 sites, 1 hour/experimentation

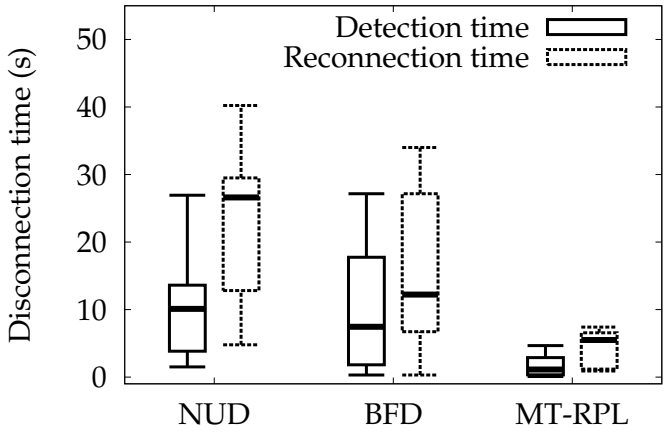
Table 5.3: Experiments parameters

measured experimental results are averaged over the 10 experiments for each unreachability detection method and site. During the experiments we have evaluated the following parameters: mobile node disconnection time from the preferred parent, percentage of packet delivery ratio (PDR) and total number of control messages sent.

We did not evaluate again the built-in mechanism for parent management from RPL, as even in simulation, it does not provide sufficient mobility support for the mobile nodes in order to be counted as an envisaged solution in any real implementation.



(a) Strasbourg site



(b) Grenoble site

Figure 5.9: Average disconnection time from parent

The disconnection time is split between detection and reconnection time as illustrated in Figure 5.9. The detection time represents the time between a mobile node receiving a broadcast packet from its preferred parent, as it will stop serving any mobile node, and the unreachability detection mechanism reacting and starting the search for a new parent. The reconnection time is the time needed to exchange RPL control messages (DIS and DIO messages) with neighboring nodes in order to reattach to the DODAG (i.e. choose a new preferred parent) plus the time needed by each unreachability detection mechanism to exchanges specific control messages until reachability is confirmed. As said before in Section 5.4.1, static nodes have periods when they

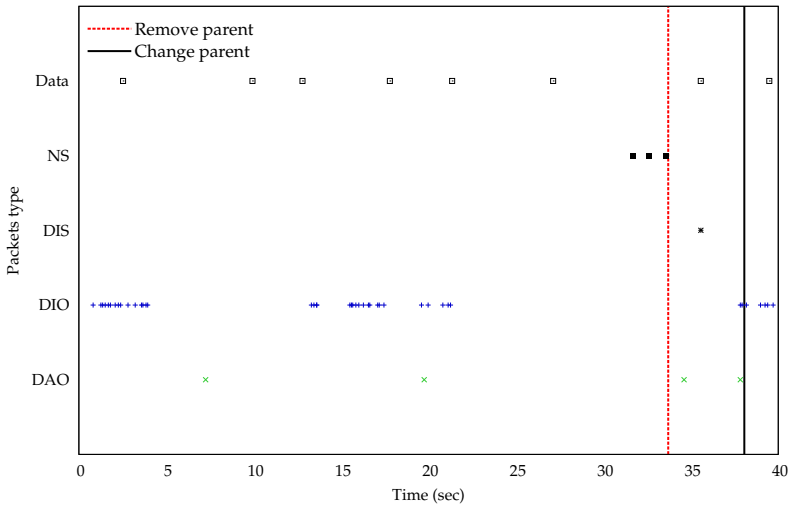
will not serve any mobile nodes, which leads to forced disconnection.

In the Strasbourg site (see Figure 5.9a), the detection time for NUD remains close to the timers defined by the protocol (i.e. 38 sec). Reconnection is close to disconnection time, as only control packets need to be exchanged before the mobile node can re-attach to the DODAG. In the Grenoble site (see Figure 5.9b), as the topology has changed, this changes also the number of nodes involved in message exchange. As the nodes are in two corridors, the channel congestion and collisions are reduced, with only the corner nodes experiencing transmissions from two sides of the corridor. Lowering these factors contributes also to lowering de detection and disconnection time, when compared to the Strasbourg site. In Figure 5.10a we represent the duration of one handover between a mobile node and a preferred parent done in the Strasbourg site with NUD. Reachability is confirmed just before the static node stops serving the mobile node. Neighbor Solicitation messages are sent by the mobile node in the network, but they are not answered. This leads to the disconnection of the mobile node from the preferred parent. The next data packet will find the mobile node without a preferred parent and RPL will send a DIS message in multicast to trigger fresh DIO messages from any node that can be chosen as parent by the mobile node. After DIO messages are received, the mobile node will choose a new preferred parent and reconnect to the DODAG.

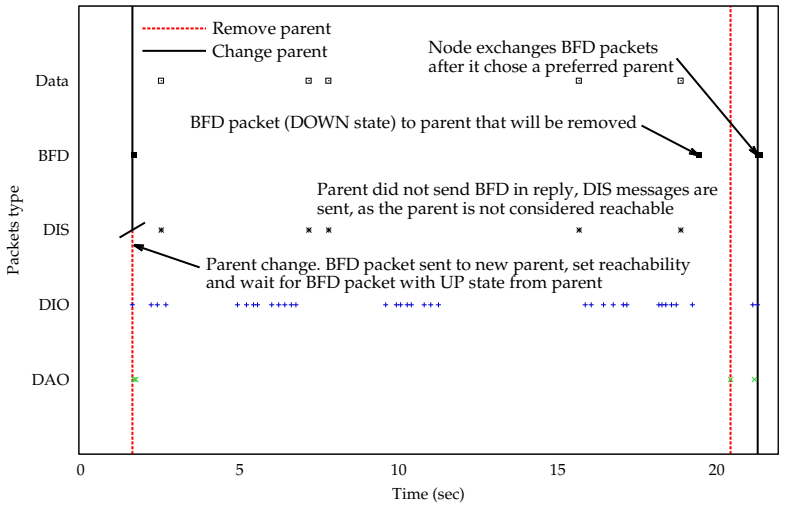
With BFD, the detection time remains roughly close to the timers set (30 sec), just like we have seen for NUD in both analyzed topologies (Strasbourg and Grenoble). After detecting the disconnection, there can be variations of the time until a node manages to regain connectivity with the DODAG. The mobile node needs to send RPL packets to search for a new parent, after which BFD control packets need to be exchanged in order to establish a BFD session. If BFD control packets do not arrive in a timely manner between nodes, delay in reconnection time can spike. In Figure 5.10b we have represented the duration of a handover in Strasbourg site done with BFD. We can see that even though the detection of disconnection from the preferred parent is done quickly (after 1.7 sec from receiving the broadcast packet announcing that the node will not reply any more to the mobile node), BFD packets are not always exchanged successfully to re-establish connectivity (this is done only after 21.35 sec since the disconnection). The mobile node waits for a BFD packet from the perspective preferred parent to check reachability. In the mean time, as the mobile node does not have any valid preferred parent, it will send DIS messages in multicast after each data packet and increase RPL control traffic, as it tries to reconnect to the DODAG. Finally, the mobile node manages to exchange BFD control packets with a static node (after 21.35 sec since the disconnection) and chooses it as a preferred parent, reattaching itself to the DODAG.

With MT-RPL, we have expected detection and reconnection times to be close to the data packet send rate, as we have seen in simulations. The experimental results, have indeed confirmed our findings from simulations. Disconnection and reconnection time do not vary between the two analyzed topology (Strasbourg and Grenoble). They stay close to the data-sending rate. In Figure 5.10c we present again the duration of a handover in the Strasbourg site done with MT-RPL. We can observe that, after the mobile node sends the whole preamble without receiving an acknowledgement, it resets its rank to infinite, remove the preferred parent and schedule a new retransmission of the preamble. This time there is an opportunistic forwarder that will acknowledge the preamble and claim the data packet from the mobile node. We can also observe a delay of 500 ms before the mobile node changes the parent. Normally the change of parent should be done instantaneously (if we would have simulated MT-RPL), but in a real implementation we have to take into consideration processing delays. The mobile node, after it receives the acknowledgement from the opportunistic forwarder will need to send the data packet, so other operations must be postponed (i.e. change of preferred parent). We chose to postpone the change of parent, as given the periodicity of data exchange (every 5 sec), there will be no negative impact on performance.

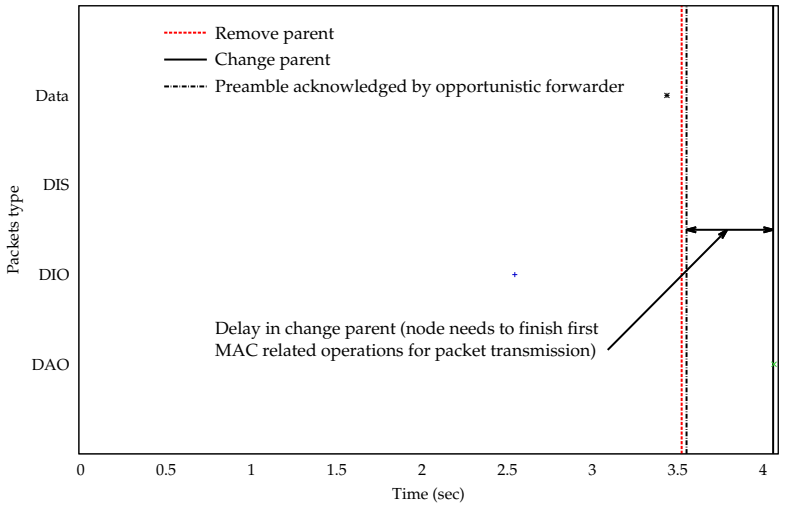
The mobile node will continuously send packets towards the root, given the constant bit rate (CBR) application that runs on the application layer, regardless if a preferred parent is set or not. The root does the same and tries to send data packets towards the mobile node using the same CBR application. This means that all unreachability detection mechanisms send the same



(a) NUD handover



(b) BFD handover



(c) MT-RPL handover

Figure 5.10: Handover between mobile and static node with external unreachability detection mechanisms; mobile node sent packets

Strasbourg site			**NUD**	**BFD**	**MT-RPL**
Mobile node to root	Packet delivery ratio	Avg. (%)	87.90	58.81	94.72
		± (%)	3.38	3.69	3.07
Root to mobile node	Packet delivery ratio	Avg. (%)	28.65	26.04	19.28
		± (%)	5.40	1.94	4.59
Grenoble site					
Mobile node to root	Packet delivery ratio	Avg. (%)	26.46	10.53	34.15
		± (%)	6.89	1.68	2.36
Root to mobile node	Packet delivery ratio	Avg. (%)	9.99	6.48	7.72
		± (%)	5.51	4.52	4.16

Table 5.4: Packet delivery ratio (PDR) with 95% confidence intervals

number of data packets from both the mobile node and the root.

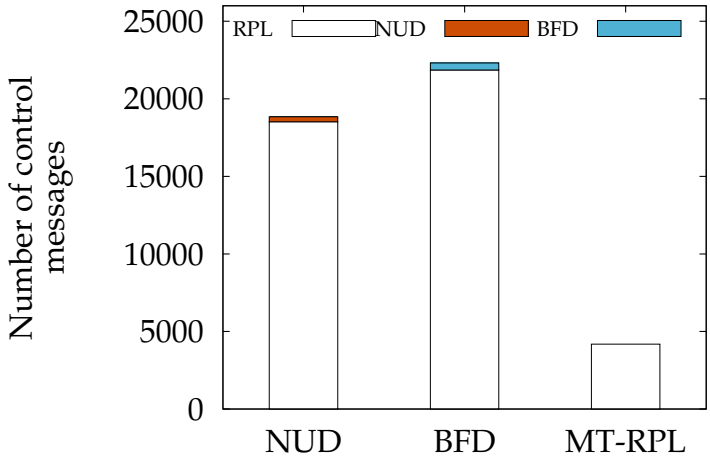
In the Strasbourg site, the PDR values for paths between the mobile node and the root are high for NUD and MT-RPL, as it can be seen in Table 5.4. Only a few packets are dropped with these two protocols. The grid topology ensures always that the preferred parent is in reach, packets traveling only 1 hop until the root (which cannot be chosen as parent as nodes would not change parents anymore). BFD on the other hand experiences an increased packet loss (61% less packets sent than MT-RPL and 49% less packets sent than NUD), as the mobile node needs to receive confirmation of reachability from the chosen parent through BFD control packets. We have previously seen in Figure 5.10b that such confirmation may not come in a timely manner, which leads to an increased packet loss. MT-RPL increases the PDR that it can provide between the mobile node and the DAG root, as it sends packets opportunistically. This also gives the mobile node more choices for a parent that will forward its data packet, lowering in the same time the disconnection time, as we have previously seen.

The paths between the root and the mobile node experiences however lower PDR that the path between the mobile node and the root. Paths between the root and mobile node need to be up-to date in order to allow packets to arrive at the mobile node. RPL is configured in storing mode for downward routes and so, nodes maintain their own routing table for all child nodes in their sub-DODAG. Using NUD and BFD as unreachability detection mechanisms, the mobile node is longer connected to the graph through the same parent and advertised routes are available more time. MT-RPL changes the parent more often. Therefore, advertising the new attachment point through DAO messages is necessary. Local conditions may lead to loss of DAO messages and so, the root find itself with an expired entry or an unreachable route to the mobile node. Paths can expire as DAO messages will not arrive after each parent change in a timely fashion, or DAO messages are lost and packets are sent from the root on a path that will never reach the mobile node. For now, we can say that the path from the root to the mobile node is still unreliable, regardless of the unreachability mechanism used.

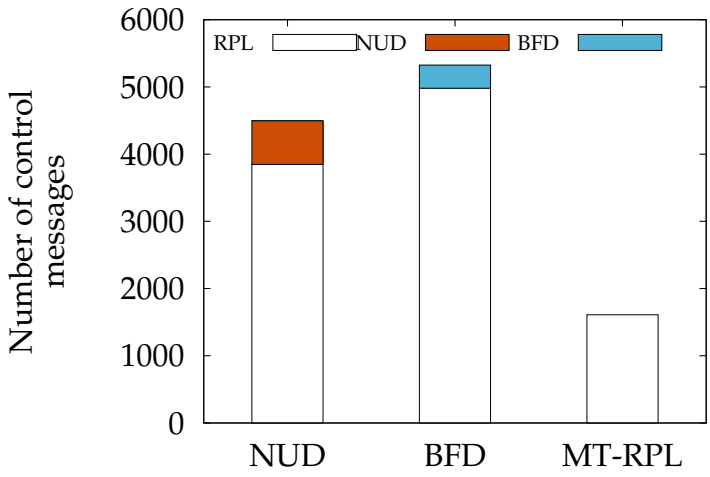
In the Grenoble site, the different topology impacts the PDR values. Even as we have comparable disconnection time as in the Strasbourg site, PDR values decrease. Analyzing why this happens, in the Grenoble site, the mobile node is always between 2 and 4 hops from the root (which explains why losses occur more on the path between the preferred parent of the mobile node and the DAG root, than between the mobile node and the preferred parent). RPL ensures multi-hop data transmission and should provide reliable forwarding. Nevertheless, as we can see in the obtained results, the longer the path, the higher is the impact of packet loss on a key parameter such as PDR. MT-RPL, as it does not introduce additional control packets in the network, doubles the PDR achieved by BFD and delivers 29% more packets than NUD between the mobile node and the root. X-Machiavel allows the mobile node to send packets opportunistically,

so static nodes closer to the sink may receive these packets, which will reduce the possibility of packet loss.

Packet collision and congestion play now an even more important role. They impact greatly the PDR values of the path from the root to the mobile node. DAO messages have now higher chances of being lost on the path from the mobile node to the root. In the Grenoble site, even more than in the Strasbourg site, the root will be deprived of up-to-date paths to the mobile node, explaining the low PDR values obtained.



(a) Strasbourg site



(b) Grenoble site

Figure 5.11: Average number of control messages sent

In order to keep the mobile node connected to the DODAG, as well as to reconnect the mobile node after a disconnection, control messages must be exchanged. The signaling overhead for each unreachability detection mechanism is presented in Figure 5.11. Enabling an external unreachability detection mechanism introduces not only new control packets, specific for each mechanism, but increases the number of sent RPL control packets in both experimental sites. RPL control packets are sent by both the mobile node and surrounding static nodes each time

the mobile node determines that connectivity has been lost with the preferred parent. Before removing the preferred parent from the parent set, a no-path DAO message is sent from the mobile node and informs the previous preferred parent that the mobile node will not send and receive packets though it anymore. After reception of the no-path DAO message, the former parent informs all nodes on the path to the route of this change. The mobile node sends no-path DAO messages, and starts searching for new parents. With NUD and BFD unreachability detection mechanisms, the mobile node will send a multicast DIS message in the neighborhood, as all parents have been deleted from the parent set. Any static node that receives the DIS message will reset its trickle timer and send new DIO messages. BFD needs to check connectivity faster than NUD (each 30 sec compared to each 38 sec) and, more importantly, needs both sides to exchange BFD control packets to ensure bi-directional connectivity. With NUD, it is more up to the mobile node to start connectivity checks and demand confirmation from the parent, the message exchange being more asymmetrical (i.e. the mobile node can send several neighbor solicit messages before receiving a neighbor advertisements message from the parent). This means that both NUD and BFD will determine the sending of more RPL control messages than MT-RPL. The difference is striking, as MT-RPL needs to send 3-5 times less control messages than BFD and 2-4 times less than NUD (depending on topology). The strong reduction of RPL control packets with MT-RPL is due to the cross-layer mechanism and to less frequent parent changes than in simulation (i.e. the mobile node has its parent in transmission range all the time, and changes of parent occur due to forced disconnection or when the preferred parent is in sleep state and another static node acts as an opportunistic forwarder). Changing the parent in MT-RPL is done with information from the MAC layer (as we have seen in Section 5.2). When a static node acts as an opportunistic forwarder for the mobile node, at RPL layer this will lead only to the change of preferred parent (eventually preceded by an exchange of DIS/DIO messages in unicast between the mobile and static nodes, if the static node is not already in the mobile node's parent set) and the sending of the according DAO message. As we can see, this will not impact the stability of the DODAG.

5.5 Conclusion

In this chapter, we have analyzed how external unreachability detection mechanisms can help RPL support mobility management. We have proposed a new cross-layer mechanism, Mobility-Triggered RPL, which leverages information from X-Machiavel MAC protocol in RPL routing protocol. MT-RPL favors medium access to mobile devices and triggers RPL operations in order to maintain efficient connectivity with the DODAG. Changes in the MAC header enable nodes operating with MT-RPL to be aware of the routing topology when they exchange packets (e.g. accept the packet at MAC layer only if the packet can make forward progress towards the DAG root). We have evaluated our main contribution in the thesis alongside NUD and BFD unreachability detection mechanisms through both simulation and experiments. From our simulation campaign, we showed that RPL built-in mechanism to mitigate mobility fails to prevent serious node disconnection, which significantly increases the packet loss. Enabling an external unreachability detection mechanism is therefore the only solution to improve mobility management performances. Results obtained from an extensive simulation campaign showed that MT-RPL significantly reduces the disconnection time, increases the packet delivery ratio while limiting the energy consumption. After we have validated our contribution through simulation, we moved on to the FIT IoT-LAb experimental platform for a real-life deployment. We also fine-tuned and improved MT-RPL (e.g. by using unicast instead of multicast DIS messages whenever possible). Deploying the unreachability detection mechanisms in two topologies, we demonstrate the robustness of our contribution, as it manages to outperform NUD and BFD in both topologies. We also saw to what extend the topology has an impact on performances. The disconnection time for the mobile node does not have great variations in both experimental sites, but the PDR values are lower in one topology than in the other due to losses along the path to the DAG root. It becomes clear that the unreachability mechanisms manage to keep the mobile

node more time connected to the DODAG, increase the PDR from the mobile node to the root, but they do not manage to keep updated also the path from the root to the mobile node. To improve the reachability between the root and the mobile node, the timing of DAO messages should be changed. A node in storing MOP waits for DAO messages from the sub-DODAG to do route aggregation, but this procedure may delay sending the DAO message from the mobile node to the root. Until the aggregated DAO message with the advertised address of the mobile node arrives at the root, the mobile node may have already changed the preferred parent and thus the route at the DAG root may no longer be valid.

This chapter concludes the contributions made in the thesis. All that remains is to draw the general conclusions of the thesis and to open perspectives for future research in the next chapter that will round up the present work.

Chapter 6

Conclusion and Perspectives

This is the final chapter of the thesis, in which conclusions are drawn. Besides this, a reminder of the addressed problems is included, alongside highlights of contributions and prospective future work.

6.1 Conclusion

At the beginning of the thesis we set the goal of providing a new communication architecture articulated around providing mobility support in WSNs. In recent years, mobility has gained an even more important role, as the need of ubiquitous connectivity drives forward the research. Maintaining connectivity with a moving node is a challenging task. A moving node has to change the point of attachment to the network as it moves, which implicates also updating the route between the mobile node and the intended destination. Achieving a seamless handover between different attachment points while maintaining connectivity is not a trivial task, as we have seen in the thesis. Mobility can be considered at each layer in the network communication stack and in this thesis we have focused on mobility management on both MAC and networking layer. In chapter 2 we have presented the state of the art of mobility management proposals at the MAC layer, while in chapter 3 we focused our attention on mobility management within the networking layer.

Our first contribution is an analysis, from a theoretical point of view, of IPv6 address auto-configuration using the optimizations for WSN proposed in Neighbor Discovery, that is included in chapter 3. The research community has been working on standards such as MIPv6 for over 10 years in an attempt to manage mobility of nodes in IP networks. MIPv6 uses extensively IPv6 address auto-configuration each time the mobile node moves to a new visited network. Our first thought was to further optimize MIPv6 for WSN requirements. In our first contribution we show that even with optimizations of Neighbor Discovery for WSN, IPv6 address auto-configuration uses larger packet sizes than what WSN networks can support (e.g. with IEEE 802.15.4 the maximum frame size sent on the medium is 127 bytes). IPv6 address auto-configuration is a key requirement for MIPv6. Fragmenting the packets leads to performance drops, as we have seen in Chapter 3. In addition, the energy consumption at the router that handles IPv6 address auto-configuration is important and shortens network lifetime. These findings have lead us to use a solution build from ground up with respects to the requirements of WSN networks, namely IPv6 Routing Protocol for Low-power and Lossy Networks (RPL). RPL has a build-in mechanism that can handle minor changes in the network, but it fails to prevent serious node disconnection due to mobility. Enhancements of RPL operations to support mobility management as well as external unreachability detection mechanisms have been proposed.

In Chapter 4, we propose our second contribution, an enhancement of RPL operations to support mobility management. Operations are localized, between the mobile node and its pre-

ferred parent and do not impact the rest of the network. The reverse trickle algorithm present in a static node is triggered when a mobile node, that is identified through a mobility flag in sent DAO messages, chooses this static node as preferred parent. This allows us to quickly detect any disconnection of the mobile node from the DODAG. We have compared in simulation our second contribution with two other state of the art proposals from the literature and manage to obtain lower control traffic overhead as well as lower disconnection time from the DODAG. Nevertheless, the mobile node needs to actively monitor the connection with the preferred parent and to synchronize to each DIO message received. The mobile node may become disconnected at the MAC layer if connectivity is checked only at the networking layer, degrading performances. In chapter 2 MAC protocols that can handle mobility management have been proposed. Leveraging information from such MAC protocols into routing protocols should thus improve mobility management.

In Chapter 5, we present our main contribution in the thesis, a cross-layer mechanism, Mobility-Triggered RPL. MT-RPL takes advantage of actions from X-Machiavel MAC protocol reported in RPL protocol to achieve mobility management. At the MAC layer, the mobile node sends its data packet with priority, either using an opportunistic static forwarder or stealing the communication channel from static nodes. These operations will be reported to RPL, which in turn takes the necessary actions and changes the preferred parent in accordance to the static node to which the mobile node sent the data packet. We evaluate our final contribution with the two other suggested unreachability detection mechanisms in RPL, NUD and BFD through both simulation and experiments. The results obtained in simulation are confirmed by experiments and point out to improvements in:

- disconnection time, which is now bound close to the data sending rate.
- control message overhead, which we found in experiments to be 2-5 times lower than obtained with NUD of BFD unreachability detection mechanisms.
- consistent high packet delivery ratio, namely on the path from the mobile node to the root.

Nevertheless, there are still some areas where further improvements could be made, such as the path between the root and the mobile node, which is still under-performing when it comes to obtained PDR values or availability of the path. In addition, another unexplored area is how MT-RPL can integrate mobile nodes that do not send any data packet, but need to be reachable from other nodes that send data to them.

It is our belief that from these three contributions, we have gained a valuable insight on how mobility management can be handled in WSN networks. We have also perspectives that have opened after we analyzed our work and which are further detailed in the next section.

6.2 Perspectives

The presented contributions in this thesis can be extended in different directions. Here are some of them.

6.2.1 Enhanced experiments

Informations about the energy consumption of the radio chipset during experiments is currently unavailable in a way that we can identify the energy spent on transmissions by each individual mechanism. We hope that support for such measurements will be quickly available in the FIT IoT-LAB experimental platform so that all researchers can benefit from a more complete analysis of their solutions.

Analyzing closely the energy consumption for transmissions (i.e. energy spend to send, receive, in sleep and in idle listening) will allow us to better understand where the energy is wasted and take actions to limit it. This will also help us extend the network lifetime.

Besides energy consumption, we envisage the usage of multiple mobile nodes in the same time. For now, only one mobile node could move. Nevertheless, this helped us validate our assumptions. The impact of multiple mobile nodes has been evaluated only in simulation, and we are confident that once there are multiple mobile nodes in the experimental platform, the performances of MT-RPL will remain at levels comparable to what we achieved now.

6.2.2 Downward route connectivity

During evaluation of downward routes, we have found that the DAG root does not always have an up-to-date route to the mobile node. Frequent reconnections of the mobile node after it changes the preferred parent need to be reported in a timely manner to the DAG root, but until now they fail to arrive timely at the root.

When multiple mobile nodes are present in the network, the problem of keeping up-to-date paths from the root to the mobile node may become more acute, as more RPL control packets need to be exchanged, increasing the overall traffic. Solutions to speed up the route propagation from nodes to their parents should be explored. A starting idea could be prioritizing DAO messages transmissions, when they come from mobile nodes, in storing MOP. In the RPL standard, nodes are expected to wait for a certain time before sending DAO messages to their parents, so that aggregation of routes can be done in storing mode. This approach can be successfully applied in static networks, where changes in topology can be handled with the built-in mechanism in RPL for parent change. Nevertheless, when mobile nodes are present, waiting to aggregate routes before sending DAO messages upward may transform a valid path to the mobile node into an expired one, as the mobile node can in the mean time change its attachment point.

6.2.3 IPv6 address auto-configuration for mobile nodes in RPL

We have seen in our theoretical analysis of IPv6 address auto-configuration with optimizations in Neighbor Discovery for WSN that this step in enabling connectivity for a node is not trivial in WSN and can lead to underachievement. The research community is working towards an IPv6 address auto-configuration standard for RPL, and some ideas have already emerged and are submitted in the form of a draft to the IETF, as we have seen in Section 3.5.

Implementing IPv6 address auto-configuration for RPL in the context of mobility is an idea worth exploring. Besides finding a new attachment point to the DODAG once it has moved, the mobile node should also be able to continue to be reachable to outside nodes (i.e. outside of the DODAG). The parents of the mobile node could have different prefixes, so keeping the same IPv6 address at the mobile node after movement becomes problematic. So we need to define new mechanisms that will allow mobile nodes to keep their IPv6 address independently of the prefix announced by the new parent (similarly to what is done in PMIPv6 or by announcing new routes towards the root with DAO messages). In addition, mobile nodes may have several IPv6 addresses at their disposal, so that they can be used similarly to how multiple addresses are used in MIPv6. These improvements should contribute to keeping connectivity between the mobile node and the network.

6.2.4 Mobility between multiple DODAGs

Until now, we allowed the mobile node to choose as preferred parent static nodes that belong to the same DODAG. The RPL standard mentions the possibility of having multiple DAGs, each one rooted at a different node. Depending on application requirements, there can be several DODAGs in the same network. As the mobile node can move inside the network, it may find itself in an area where all neighboring nodes belong to a different DODAG than the one the mobile node was previously attached.

Exploring how a mobile node can change DODAGs without introducing changes in the already built topology is an interesting perspective. It will allow a mobile node to remain connected with

nodes outside the DODAG, but it will also pose significant challenges on how to transfer any connections to the mobile node from one DAG root to another.

Appendix A

Résumé thèse en français

A.1 Abstract

Dans les derniers temps, des nouveaux types de réseaux, qui interconnecte des capteurs avec différentes capacités de communication et l'Internet, ont émergé. Tous ces réseaux forment ce qui nous appelons L'Internet des Objets. Une partie importante de l'Internet des Objets sont les réseaux des capteurs sans fil (WSN - Wireless Sensor Network). Les capteurs communiquent aves des ondes radio, ce qui donne des nouveaux défis pour la communication (exemple : des liens instables). Au cas oú, même si les noeuds bougent, ils s'attachent ou se détachent du réseau, ils doivent rester joignables par des autres noeuds. Des nouveaux applications développés pour WSN nécessitent des noeuds mobiles (exemple : suivi des patients). Il faut donc int'egrer ces noeuds mobiles dans les réseaux et offrir support pour la communication. L'objectif de cette thése est de définir une nouvelle architecture de communication articulée autour de la mobilité dans les réseaux de capteurs sans fil.

Dans les réseaux IP (Internet Protocol), le support pour mobilité est offert par des protocoles comme Mobile IPv6 (MIPv6). Dans les WSN, on ne trouve pas beaucoup de solutions basées sur MIPv6 et celui qui existe ne donne pas des résultats concluants. Dans cette contexte, notre premiére contribution est d'analyser plus proche si MIPv6 peut être considéré faisable dans WSN. Nous avons fait une étude théorique de l'impact de l'auto-configuration d'adresse IPv6, avec les optimisations de protocole Neighbor Discovery (ND) pour WSN, sur les performances des réseaux de capteurs sans fil. L'auto-configuration d'adresse IPv6 est une étape nécessaire avant que des opérations MIPv6 pour la gestion de mobilité puissent être démarrée. Notre analyse découvre que la taille trop élevée des messages nécessaires dans ND et l'énergie dépensée par les routeurs pendant les opérations d'auto-configuration d'adresses. Elle met aussi en question la future intégration de MIPv6 dans les réseaux des capteurs sans fil. Nous avons décidé de focaliser notre attention sur RPL, un protocole de routage IPv6 qui a était développé spécifiquement pour les réseaux de capteurs sans fil. RPL a déjà intégré un mécanisme pour la gestion des parents, mais le mécanisme réagissait lentement aux changements de la topologie et il échouait à prévenir la déconnexion des noeuds mobiles. Pour cette raison, des améliorations des opérations du RPL on était proposés pour mieux supporter les noeuds mobiles ou l'utilisation des mécanismes externes de détection de l'inaccessibilité.

Notre deuxiéme contribution est une proposition des améliorations des opérations RPL pour mieux supporter les noeuds mobiles. Nous proposons un algorithme trickle inverse pour les noeuds statiques qui joue le rôle de parent préféré des noeuds mobiles et l'annonce du statut de mobilité par le noeud mobile. Nous avons réussi à diminuer le trafic de contrôle et le temps de déconnexion du noeud mobile de DODAG ainsi que d'augmenter le taux de succś des paquets envoyés en comparaison avec des autres solutions dans la littérature. Même si nous avons réussi a réduire le temps de déconnexion, le noeud mobile doit encore rester en contact permanent avec

son parent (il doit se synchroniser avec les messages DIO reçus). Nous avons donc continué à chercher une meilleure solution. Ainsi, nous avons utilisé un protocole d'accés au medium qui tient compte de la mobilité et annonce le protocole de routage des changements apparus pour mieux gérer la mobilité du noeud.

Finalement, on propose une derniére contribution, un mécanisme inter-couche entre le protocole MAC X-Machiavel et le protocole RPL qui améliore la gestion de mobilité dans RPL. L'idée derrière notre contribution principale est de déclencher actions dans RPL (exemple : changement du parent préféré), quand des évènements au niveau de la couche accés au medium ont lieu (exemple : transmissions des paquets depuis le noeud mobile vers les noeuds statique). Avec des interconnexions entre les différentes couches, nous poussons en avant la structure de communication basée sur des couches indépendantes et nous arrivons aux améliorations vis à vis des autres mécanismes de détection de l'inaccessibilité suggérée par RPL. Notre contribution principale limite le temps de déconnexion proche du cadence d 'envoie des paquets et améliore le taux de réussite. Les couches liaison et routage travaillent maintenant ensemble, ce qui détermine aussi une baisse du trafic de control.

A.2 Introduction et contexte

Le premier usage des capteurs est lié au projet de l'armée américaine pour la détection de sous-marines soviétique dans le années 1950. Le réseau Sound Sourveillance System (SOSUS) était composé de capteurs acoustiques immergés dans les océanes Atlantique et Pacifique et est encore en service aujourd'hui. Des déploiements comme cella étaient très couteuses et se focalisaient sur la détection d'évènement, tout en négligeant la partie réseau. Néanmoins, une fois que les coûts de production ont baisé (les semi-conducteurs deviennent moins chers), le monde des réseaux de capteurs a gagné en attractivité. Pendant cette thèse nous nous focaliserons sur le fonctionnement des protocoles réseaux.

A.2.1 Structure des réseaux sans fil

La communication entre diffèrent capteurs sans fil se fait utilisant une pile protocolaire qui ressemble à la pile de communication OSI (Open Systems Interconnection), pour assurer la intercommunication entre les WSN et les réseaux IP. Les couches standardisées dans WSN sont définies par le standard IEEE 802.15.4 pour les couches physique et accés au medium. Après, nous avons la couche réseau qui est divisé entre la couche d'adaptation 6LoWPAN (qui fait de la compression des entêtes IP), l'adressage (IPv6) et routage (utilisant RPL - Ipv6 Routing Protocol for Low-Power and Lossy Networks). La couche supérieure utilise UDP pour le transport des paquets IP et la couche applicative utilise des règles établies par le protocole IEFT CoAP pour communiquer.

A.2.2 Motivation

Des WSN sont déployés majoritairement avec des capteurs statiques, mais les derniéres évolutions utilisent de plus en plus des capteurs mobiles (pour surveiller des patients ou pour le suivi des cibles dans les champs des batailles). La gestion de ces capteurs, tout en gardant la connectivité avec le réseau, est difficile et un transfert transparent pour l'application entre des points différents d'attachement au réseau n'est pas encore fiable. L'idée de cette thèse est de développer des solutions efficaces pour la gestion de la mobilité tout en gardant la connectivité durant le mouvement. Nous nous focaliserons sur la définition des solutions de gestion de la mobilité en utilisant des informations données par la couche routage. Nous avons choisi un protocole de routage qui support un noeud mobile et qui peut construire des connexions bidirectionnelles.

A.3 La gestion de la mobilité dans la couche réseau

Au sein de la couche réseau on peut distinguer deux types de mobilité : roaming et handover. Les services qui sont offerts dans une autre endroit de l'endroit oú ils étaient enregistrés et les noeuds restent connectés au réseau, sont dans un processus de roaming. Si les noeuds changent leur point d'attachement au réseau, sans perdre leur connectivité, ils font un handover.

La gestion de la mobilité est faite pendant toute cette thèse quand des handovers ont lieu. La mobilité physique des noeuds ainsi que les capacités limités de communication déterminent un changement du point d'attachement. Nous examinerons deux pistes possibles : l'adaptation des protocoles comme Mobile IPv6 dans les WSN et l'utilisation d'un protocole de routage, RPL, qui peut supporter la mobilité.

Dans la littérature, les adaptations de MIPv6 dans WSN ont rencontré des difficultés pendant la détection de mouvement (détecter un handover), une étape importante avant que des opérations de gestion de la mobilité peuvent être faites. Nous avons essayé de surmonter ces difficultés par l'utilisation des derniéres optimisations du protocole Neighbor Discovery (ND) vis à vis de l'auto-configuration des adresses IPv6. Si on arrive à obtenir une nouvelle adresse IPv6 dans une façon efficace, on peut accélérer la phase de détection du mouvement dans MIPv6.

A.3.1 L'auto-configuration des adresses IPv6

Neighbor Discovery est le protocole de choix pour faire l'auto-configuration des adresses IPv6 mais aussi pour la résolution d'adresse, la détection de duplicata d'adresse, la détection de routeurs sur la liaison, la détection d'inaccessibilité ou la redirection des paquets dans les réseaux IP. Si on veut utiliser les mêmes fonctions dans WSN, il faut faire des changements dans ND, car ce protocole ne supporte pas des liens transitifs (les liens sans fil peuvent être asymétriques et deux voisins sans fil peuvent faire partie des réseaux IPv6 différentes). ND utilise suivant des transmissions multicast (dans les WSN, ce type de transmission n'est pas souvent utilisé, car les noeuds consomment beaucoup d'énergie). Pour cella, des extensions du ND ont été faites pour l'utilisation dans WSN : des messages de contrôle, comme Router Advertisement sont envoyés en unicast quand un Router Solicitation est reçu ; la détection de duplicata d'adresse est éliminé une fois qu'on utilise des adresses IPv6 construites suivant EUI-64 ; l'enregistrement des adresses IPv6 élimine le besoin d'envoie des messages Neighbor Solicitation en multicast. Un noeud qui veut auto-configurer une adresse IPv6 avec les optimisations de ND créera une adresse liaison locale et enverra un message Router Solicitation en muticast vers des routeurs dans le voisinage. Un router répondra à la sollicitation avec des informations nécessaires au noeud pour configurer une adresse IPv6 globale. Une fois l'adresse globale obtenue, le noeud essayera de l'enregistrer au router (avec un message Neighbor Solicitation en unicast) et attendra le message de confirmation de la part de routeur. Nous avons analysé d'un point de vue théorique combien de temps cette procédure dure et quel est l'impact sur les performances du réseaux. Le réseau analysé a des caractéristiques typiques rencontrées dans beaucoup de déploiements du WSN. Il utilise des protocoles standard pour communiquer, qui étaient évoqués dans la structure des réseaux sans fil. Après l'analyse, nous avons constaté que la taille des options dans le message Router Solicitation est trop grande pour être envoyé dans des cadres IEEE 802.15.4 même avec la compression 6LoWPAN. Les routers impliqués dans des opérations lies au ND consomment très vite leur énergie disponible, car ils sont impliqués dans toutes les transmissions, ce qui crée aussi un point de défaillance unique dans le réseau. L'utilisation de l'auto-configuration des adresses IPv6 est encore loin d'être prêt à l'usage dans les WSN, ainsi que l'utilisation des protocoles de gestion de la mobilité come MIPv6, qui est mise en question dans des WSN. Pour cela, nous avons envisagé des autres pistes de recherches orientées vers l'utilisation d'un protocole de routage développé spécifiquement pour les WSN, c'est-à-dire RPL.

A.3.2 RPL - Standard IPv6 Routing Protocol for Wireless Sensor Networks

RPL est un protocole de routage basé sur des vecteurs distance qui construisent un arbre de routage orienté vers la racine (DODAG). Il définit des nouveaux messages de contrôle ICMPv6 qui sont utilisés au début pour construire la topologie et après pendant sa maintenance. Le DIO - DODAG Information Object - contient toutes les informations nécessaires pour que les noeuds puissent se connecter au DODAG. Le DIO est envoyé selon l'algorithme trickle. Une fois un DIO réceptionné, le noeud peut calculer sa position relative dans l'arbre (donné par le rang) et peut choisir un parent préféré. Le DIS - DODAG Information Solicitation - est envoyé par les noeuds qui veulent des informations vis à vis des potentielles DODAG dans leur voisinage. Les noeuds qui réceptionnent des DIS répondent avec des DIO. Le DAO Destination Advertisement Object - est envoyé pour annoncer l'adresse du noeud à la racine ou au parent préféré. RPL était envisagé à l'utilisation dans les WSN avec noeuds statiques, mais RPL peut également accommoder des noeuds mobiles. Il y a quand même des problèmes dans la gestion du parent qui peut laisser un noeud mobile déconnecté du réseau. RPL permet de changer le parent seulement si un DIO qui annonce un rang mieux que le rang du parent est reçu. Si le noeud mobile est dans une région du réseau oú toutes les noeuds annoncent des rang supérieurs, il deviendra déconnecté. Pour éviter ce situation, RPL propose d'utiliser des mécanismes de détection de l'inaccessibilité (tel que Neighbor Unreachability Detection, Bidirectional Forwarding Detection ou Hints from lower layers via Layer 2 triggers) et la communauté scientifique a proposé des améliorations des opérations du RPL pour gérer mieux la mobilité des noeuds.

A.3.3 Mécanismes de détection de l'inaccessibilité

Neighbor Unreachability Detection (NUD) fait partie du protocole Neighbor Discovery et il est utilisé pour vérifier l'état de communication entre deux voisins. Si les voisins sont joignables, c'est-à-dire ils ont échange en préalable des messages de contrôle, ils peuvent échanger entre eux des paquets pour un certain période de temps avant que l'état de la liaison doive être de nouveau vérifié. Quand les messages de contrôle ne puissent pas être échangés ' il n'y a pas des réponses de l'autre part, les voisins sont considérés inaccessibles. Dans ce cas, RPL commencera à chercher aussi un nouveau parent pour le noeud.

Bidirectional Forwarding Detection (BFD) emploie le même principe come NUD et échange des messages de contrôle pour vérifier si deux voisins sont joignables, mais il y a des différences. BFD continue à échanger des messages de contrôle même si les voisins sont déclarés joignables et ces messages sont envoyés encapsulés comme des messages UDP, pas comme des messages ICMPv6 envoyés par NUD.

Les L2 triggers offre juste un cadre général pour la communication inter-couche en utilisant des primitives. Pour chaque couche accés au medium, une implémentation spécifique doit être faite.

A.3.4 Amélioration des opérations du RPL pour gérer la mobilité des noeuds

Dans la littérature de spécialité, des propositions d'amélioration des opérations du RPL ont été proposées. Une de ces propositions enlève le trickle timer pour l'envoie des DIO et elle utilise un temporisation statique. Une autre proposition est de faire la gestion d'envoie des DIS dynamique, en fonction de la mobilité du noeud (donnée par le nombre des changements des parents dans un intervalle d'observation).

Nous avons fait aussi deux propositions pour la gestion de la mobilité avec l'aide du RPL : une amélioration des opérations du RPL pour gérer mieux la mobilité des noeuds et un nouveaux protocole de communication inter-couche, qui s'appelle Mobility Triggered-RPL, qui prend en compte des actions prises par la couche accés au medium dans le protocole de routage RPL.

A.4 Amélioration des opérations du RPL pour gérer la mobilité des noeuds

La deuxième contribution dans cette thèse est une amélioration des opérations du RPL pour gérer mieux la mobilité des noeuds. Nous sommes partie de la prémisse que le noeud mobile est identifié par un indicateur dans les DAO envoyés. Ainsi, il peut s'attacher au DODAG juste comme une feuille. Le noeud mobile, une fois attaché à un parent, restera long temps attaché. Les caractéristiques de notre proposition sont : elle est localisée, avec des interactions seulement entre le parent et que on utilise un trickle timer inverse. On peut détecter l'inaccessibilité des deux côtés : du côté du parent quand à la fin de trickle timer inverse il ne reçoit pas des nouvelles DAO du part du noeud mobile et du coté du noeud mobile si un seuil (donné par le nombre des messages DIO raté) statique est dépassé. Le trickle timer inverse est lancé sur le parent quand il reçoit un DAO d'un noeud mobile. Au début, on considère un intervalle large entre deux DIO consécutifs, après on divise par deux cet intervalle jusqu'a un intervalle minimal. Quand on a atteint l'intervalle minimal on demande aux noeuds de sous-arbre d'envoyer un nouveau DAO. Si le noeud mobile arrive hors de la portée de son parent, le dépassement de seuil entre DIO annoncera l'inaccessibilité du parent. Le noeud mobile doit alors, à ce moment, de chercher un autre parent dans son voisinage.

Cette solution a été comparée avec autres solutions dans la littérature par simulation. Les résultats montrent que notre solution limite le temps de déconnexion du noeud mobile au seuil fixé. La solution est indépendante de métrique et de couche accés au medium utilisé et elle n'utilise pas des messages de contrôle externe. Elle est déployée juste entre le noeud mobile et son parent et elle entraine des modifications minimales dans RPL. Néanmoins, le noeud mobile doit surveiller la liaison avec son parent en permanence. Le taux de réussite pour le chemin entre le noeud mobile et le puits est faible à cause de temps long de détection si on utilise une solution de gestion de la mobilité juste dans la couche réseau.

A.5 Mobility-Triggered RPL

Alors, nous avons continué à chercher une solution qui peut améliorer les performances du noeud mobile. Nous nous sommes retournés vers l'utilisation de la communication inter-couche, entre la couche accés au medium et la couche réseau. Pour la couche accés au medium (MAC), entre la pléiade des protocoles disponibles, les protocoles MAC à préambule sont les plus adaptés a intégrer un noeud mobile. Ils offrent aussi une efficacité énergétique élevée. Le protocole X-Machiavel était développé spécifiquement pour favoriser les transmissions des noeuds mobiles sur les transmissions des noeuds statiques. Nous relierons ce protocole au RPL avec des triggers, ce que va constituer MT-RPL.

Avant de présenter comme MT-RPL fonctionne, nous ferons une présentation de X-Machiavel (le protocole MAC à préambule), pour mieux comprendre le fonctionnement du MT-RPL. X-Machiavel est un protocole MAC à préambule qui favorise la transmission des données qui appartiennent au noeud mobile dans n'importe quelles conditions du trafic dans le réseau. Le principe du fonctionnement du protocole X-Machiavel est basé sur la transmission d'un message de contrôle, nommé préambule, avant la transmission du paquet des données. Ce préambule est envoyé dans plusieurs transmissions, un strobe à la fois. Entre deux strobes, le destinataire du préambule et implicitement du paquet des données, peut répondre et acquitter le préambule. Ainsi, le noeud source peut continuer avec la transmission du paquet des données. Durant la transmission des données, le noeud mobile peut rencontrer différents états du canal de communication.

Quand le canal de communication n'est pas utilisé, le noeud mobile envoie des strobes et il attend l'acquittement du préambule avant la transmission du paquet des données. Si dans le voisinage il y a un noeud statique qui écoute la transmission, ce noeud statique peut acquitter le préambule du noeud mobile, même si il n'est pas le destinataire du préambule. Si le noeud mobile reçoit un acquittement qui n'est pas venu de la destination initiale souhaitée, il changera la destination du paquet des données vers l'adresse du nouveau noeud statique, qui est prêt à

recevoir le paquet des données. Ainsi, le noeud mobile réussit à transmettre son paquet des données, même si la destination initiale n'est plus dans le voisinage.

Si le noeud mobile est dans une zone oú le canal de communication est occupé et il a un paquet des données à transmettre, le noeud mobile écoutera le canal. Les noeuds statiques qui ont des paquets à transmettre enverront des préambules à leur part. Dans ce préambule, les strobes informent les noeuds dans leur voisinage si la source de ce strobes accepte ou pas de donner son canal de communication aux noeuds mobiles qui pourraient être dans leur voisinage. Quand un noeud mobile entend un strobe qui permet de voler le canal de la communication au noeud statique, il va mettre à jour la destination du paquet des données et il va les transmettre au noeud statique qui a envoyé le strobe. Une fois que le noeud statique reçoit un paquet qui vient d'un noeud mobile, il jouera le rôle de forwarder (il retransmettra les paquets vers la destination finale de la part du noeud mobile). Pour transmettre le paquet du noeud mobile, le noeud statique changera le type de préambule envoyé à un type qui ne permet plus aux autres noeuds de voler son canal de communication, même si des noeuds mobile restent encore dans son voisinage. Après ce préambule est acquitté, le noeud statique transmet le paquet du noeud mobile. Il ne faut pas oublier qu'au départ le noeud statique avait son propre paquet des données à envoyer. Pour ne pas perdre l'occasion d'envoyer son paquet des données, après l'envoie du paquet du noeud mobile, le noeud statique continue à envoie des préambules qui ne permettent pas aux autres noeuds de voler le canal de communication. Après l'acquittement du préambule, le noeud statique réussit de transmettre aussi son propre paquet des données.

Des triggers seront envoyés depuis X-Machiavel vers RPL après chaque action liée au noeud mobile. Ces triggers sont à la base de MT-RPL. Cette fois on prend en compte aussi la position de chaque noeud dans le graph, le rank. Le rank sera envoyé par tous les noeuds dans leurs préambules au niveau accés au medium. Cette information aidera les noeuds mobiles ainsi que les noeuds statiques à déterminer si ils peuvent participer ou non à la communication en cours.

Les noeuds statiques, acquitteront des préambules envoyés par des noeuds mobiles seulement si leur rank est inferieur au rank présent dans le préambule (les noeuds statiques sont plus proches du racine).

Les noeuds mobiles à leur part, si ils envoient tout le préambule et il n'y a aucune acquittement, ils sont dans une partie du réseau oú toutes les autres noeuds ont des rank supérieurs à leur rank. Ainsi, ils sont incapables d'envoyer leurs paquets des données. Pour ne pas perdre la connectivité avec le graph, les noeuds mobiles changeront leur rank à infini pour la prochaine retransmission. Ça leur permettra d'avoir leur préambule acquitté par n'importe quel autre noeud statique dans leur voisinage, car tous les noeuds auront maintenant un rank inferieur au rank publié dans le préambule.

Au moment qu'un noeud statique (le forwarder) acquitte le préambule du noeud mobile, cette information est remontée par des triggers dans RPL. La destination du paquet est changée aussi au niveau routage. Si le paquet est réceptionné par le forwarder (soit le forwarder acquitte les données, soit le noeud mobile reçoit un préambule qui ne permet plus de voler le canal de la part du forwarder), le parent du noeud mobile au niveau RPL devient le forwarder.

Pour vérifier les performances de MT-RPL nous avons effectué des campagnes des simulations ainsi que des expérimentations dans la plateforme FIT IoT-LAB. MT-RPL, qui est un protocole inter-couche entre la couche accés au medium et la couche réseau, a été comparé avec NUD et BFD. NUD gère la connectivité du noeud mobile entièrement à la couche réseau. BFD lie la couche transport à la couche réseau pour assurer la connectivité du noeud mobile. NUD et BFD ne sont pas capables de maintenir la connectivité du noeud mobile. Avec ces mécanismes, la perte des paquets des données est importante. Avec MT-RPL nous avons réussi de synchroniser les informations des deux couches (la couche accés au medium et la couche réseau), ce qui réduit le temps de déconnexion du noeud mobile et augmente le taux de réussite pour les paquets des données. Pendant les expérimentations nous avons également observé l'impact sur la performance des différentes topologies. Si près de noeud mobile il y a peu de parents potentiels ou le chemin vers la racine est rallongé, les performances vis à vis du taux de réussite baissent.

A.6 Conclusions

Au début de cette thèse, le but était de concevoir une nouvelle architecture de communication qui peut supporter des noeuds mobiles dans les réseaux des capteurs sans fil. Dans les derniéres années, la mobilité dans les réseaux des capteurs sans fil a gagné un rôle plus important. Rester connecté pendant le mouvement et changer le point d'attachement au réseau sans perdre la connectivité n'est pas triviale, comme nous avons vu pendant cette thèse. La mobilité peut être considérée à chaque couche de communication et pendant cette thèse nous nous somme concentrés sur la mobilité dans les couches accés au medium et réseau.

La première contribution analyse d'un point de vue théorique l'auto-configuration des adresses IP dans un réseau des capteurs sans fil en utilisant les optimisations pour WSN proposés dans Neighbor Discovery. Nous avons essayé d'optimiser encore MIPv6. Notre analyse a montré qu'en utilisant les optimisations proposées dans Neighbor Discovery, l'auto-configuration des adresses IPv6 utilise des paquets d'une taille trop grande pour WSN. Ainsi, même si on fragmente les paquets, la performance baisse. La consommation d'énergie pour les routers qui gèrent l'auto-configuration est importante et elle réduit la durée de vie du réseau.

Nous nous sommes retournés vers autres solutions, notamment l'utilisation du protocole RPL, qui a intégré un mécanisme de gestion des changements dans le réseau, mais il ne peut pas gérer la mobilité des noeuds. La deuxième contribution améliore les opérations du RPL pour gérer la mobilité. Les opérations sont faites localement, entre le noeud mobile et son parent. Cella n'impacte pas le reste du réseau. Le trickle inverse, qui est présent dans le noeud statique, est lancé une fois que le noeud mobile choisit comme parent le noeud statique. Avec ce trickle inverse on peut détecter plus rapidement les déconnexions, comme nous avons vu pendant les simulations. Quand même, nous pourrions améliorer encore les performances si nous prenons en compte aussi des informations venues de la couche accés au medium.

Avec MT-RPL nous prenons en comptes des informations données par le protocole X-Machiavel dans RPL. Comme dans la couche accés au medium le noeud mobile peut envoyé ses données avec priorité, on va rapporter ces opérations aussi au RPL. Ce protocole a été comparé avec NUD et BFD dans simulations et expérimentations. Les résultats montrent que MT-RPL réduit le temps de déconnexion du noeud mobile, il limite les messages de contrôle envoyés et il a un taux de réussite élevé depuis le noeud mobile vers la racine. Néanmoins, on doit encore améliorer les performances du chemin de la racine vers le noeud mobile.

A.7 Perspectives

Nous avons identifié trois perspectives pour le futur :

- Amélioration du chemin depuis la racine vers le noeud mobile, par la priorisation du messages DAO envoyés depuis le noeud mobile et l'implémentation de l'auto-configuration IPv6 dans RPL pour cacher le mouvement du noeud mobile à la racine.
- Expérimentations détaillées pour analyser la consommation d'énergie pendant les transmissions des données. Plusieurs noeuds mobile sont envisagés à être déployés dans la plateforme expérimentale FIT IoT-LAB et on peut vérifier aussi la robustesse des propositions quand le numéro des noeuds mobiles augmente.
- Mobilité entre différents DODAGs. La transiton entre DODAGs doit être faite sans perte de performance pour le noeud mobile.

Bibliography

[802a] *IEEE 802.15 WPAN Task Group*. http://www.ieee802.org/15/pub/TG4.html.

[802b] "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". In: *ANSI/IEEE Std 802.11, 1999 Edition (R2003)* (2003), pp. i–513.

[A8] *A8 Open Node*. https://www.iot-lab.info/hardware/a8/.

[AKK04] J.N. Al-Karaki and A.E. Kamal. "Routing techniques in wireless sensor networks: a survey". In: *IEEE Wireless Communications* 11.6 (2004), pp. 6–28.

[Aky+02] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "Wireless sensor networks: a survey". In: *Computer Networks* 38.4 (2002), pp. 393 –422. URL: http://www.sciencedirect.com/science/article/pii/S1389128601003024.

[Ali+05] M. Ali, T. Suleman, and Z.A. Uzmi. "MMAC: a mobility-adaptive, collision-free MAC protocol for wireless sensor networks". In: *24th IEEE International Performance, Computing, and Communications Conference (IPCCC)*. 2005, pp. 401–407.

[All+07] J. Allred, A. B. Hassan, S. Panichsakul, W. Pisano, P. Gray, and J. Huang. "Sensorflock: an airborne wireless sensor network of micro-air vehicles". In: *ACM SenSys*. 2007.

[AP14] N. Accettura and G. Piro. "Optimal and secure protocols in the IETF 6TiSCH communication stack". In: *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*. 2014, pp. 1469–1474.

[AY05] K. Akkaya and M. Younis. "A Survey on Routing Protocols for Wireless Sensor Networks". In: *Ad Hoc Networks, Elsevier* 3.3 (2005), pp. 325 –349.

[Ba+11] P.D. Ba, B. Gueye, I. Niang, and T. Noel. "MoX-MAC: A low power and efficient access delay for mobile wireless sensor networks". In: *2011 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC)*. 2011, pp. 1–6.

[Bha+07] D. Bhatia, L. Estevez, and S. Rao. "Energy Efficient Contextual Sensing for Elderly Care". In: *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*. 2007, pp. 4052–4055.

[Bra+10] A. Brandt, J. Buron, and G. Porcu. *Home Automation Routing Requirements in Low-Power and Lossy Networks*. RFC 5826. IETF, 2010.

[Bue+06] M. Buettner, G.V. Yee, E. Anderson, and R. Han. "X-MAC: A Short Preamble MAC Protocol for Duty-cycled Wireless Sensor Networks". In: *ACM International Conference on Embedded Networked Sensor Systems (SenSys)*. Boulder, Colorado, USA, 2006.

[CD98] A. Conta and S. Deering. *Generic Packet Tunneling in IPv6 Specification*. IETF RFC 2473. 1998.

[Cer+09] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. "Monitoring Heritage Buildings with Wireless Sensor Networks: The Torre Aquila Deployment". In: *IPSN*. IEEE, 2009.

[Cha+08] X. Chao, W. Dargie, and G. Lin. "Energy Model for H2S Monitoring Wireless Sensor Network". In: *11th IEEE International Conference on Computational Science and Engineering (CSE)*. 2008, pp. 402–409.

[Che+09] M. Cheng, M. Kanai-Pak, N. Kuwahara, H. I. Ozaku, K. Kogure, and J. Ota. "Dynamic Scheduling Based Inpatient Nursing Support: Applicability Evaluation by Laboratory Experiments". In: *Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems*. Casemans '09. Nara, Japan: ACM, 2009, pp. 48–54. URL: http://doi.acm.org/10.1145/1538864.1538873.

[Che+12] Y. Chen, J.P. Chanet, and K.M. Hou. "RPL Routing Protocol a case study: Precision agriculture". In: *First China-France Workshop on Future Computing Technology (CF-WoFUCT 2012)*. Harbin, China, Feb. 2012. URL: https://hal.archives-ouvertes.fr/hal-00681319.

[Cho+08] S.C Choi, J. W. Lee, and Y. Kim. "An Adaptive Mobility-Supporting MAC Protocol for Mobile Sensor Networks". In: *IEEE Vehicular Technology Conference, VTC Spring*. 2008, pp. 168–172.

[CK03] C. Y. Chong and S. P. Kumar. "Sensor Networks: Evolution, Opportunities, and Challenges". In: *Proceedings of the IEEE* 91.8 (2003), pp. 1247–1256.

[Con+06] A. Conta, S. Deering, and M. Gupta. *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*. RFC 4443. IETF, 2006.

[CT10] Y. Chen and A. Terzis. "On the Mechanisms and Effects of Calibrating RSSI Measurements for 802.15.4 Radios". In: *Wireless Sensor Networks*. Ed. by JorgeSá Silva, Bhaskar Krishnamachari, and Fernando Boavida. Vol. 5970. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, pp. 256–271. URL: http://dx.doi.org/10.1007/978-3-642-11917-0_17.

[Dag+07] S. Dagtas, Y. Natchetoi, and H. Wu. "An Integrated Wireless Sensing and Mobile Processing Architecture for Assisted Living and Healthcare Applications". In: *Proceedings of the 1st ACM SIGMOBILE International Workshop on Systems and Networking Support for Healthcare and Assisted Living Environments*. HealthNet '07. San Juan, Puerto Rico: ACM, 2007, pp. 70–72. URL: http://doi.acm.org/10.1145/1248054.1248074.

[DC+03] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. "A High-throughput Path Metric for Multi-hop Wireless Routing". In: *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. MobiCom '03. San Diego, CA, USA: ACM, 2003, pp. 134–146. URL: http://doi.acm.org/10.1145/938985.939000.

[DD13] Q. Dong and W. Dargie. "A Survey on Mobility and Mobility-Aware MAC Protocols in Wireless Sensor Networks". In: *IEEE Communications Surveys Tutorials* 15.1 (2013), pp. 88–100.

[Dem+06] I. Demirkol, C. Ersoy, and F. Alagoz. "MAC protocols for wireless sensor networks: a survey". In: *IEEE Communications Magazine* 44.4 (2006), pp. 115–121.

[Dev+11] P. Devis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. *The Trickle Algorithm*. RFC 6206. IETF, 2011.

[Doh+09] M. Dohler, T. Watteyne, T. Winter, and D. Barthel. *Routing Requirements for Urban Low-Power and Lossy Networks*. RFC 5548. IETF, 2009.

[DP10] W. Dargie and C. Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. Wiley Publishing, 2010.

[Du+10] W. Du, F. Mieyeville, and D. Navarro. "Modeling Energy Consumption of Wireless Sensor Networks by SystemC". In: *2010 Fifth International Conference on Systems and Networks Communications (ICSNC)*. 2010, pp. 94–98.

[Dun+07] A. Dunkels, F. Osterlind, N. Tsiftes, and Z He. "Software-based Sensor Node Energy Estimation". In: *Proceedings of ACM SenSys*. 2007.

[Dun03] A. Dunkels. "Full TCP/IP for 8-bit Architectures". In: *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*. MobiSys '03. San Francisco, California: ACM, 2003, pp. 85–98. URL: http://doi.acm.org/10.1145/1066116.1066118.

[Fit] *Future Internet (FIT) - Internet of Things testbed*. http://fit-equipex.fr/.

[Fot+12] H. Fotouhi, M. Zuniga, M. Alves, A. Koubaa, and P. Marrón. "Smart-HOP: A Reliable Handoff Mechanism for Mobile Wireless Sensor Networks". In: *Proceedings of the 9th European Conference on Wireless Sensor Networks*. EWSN'12. Trento, Italy: Springer-Verlag, 2012, pp. 131–146. URL: http://dx.doi.org/10.1007/978-3-642-28169-3_9.

[Fot+15] H. Fotouhi, D. Moreira, and M. Alves. "mRPL: Boosting mobility in the Internet of Things". In: *Ad Hoc Networks* 26 (2015), pp. 17 –35. URL: http://www.sciencedirect.com/science/article/pii/S157087051400225X.

[Gad+14] O. Gaddour, A. Koubaa, R. Rangarajan, O. Cheikhrouhou, E. Tovar, and M. Abid. "Co-RPL: RPL routing for mobile low power wireless sensor networks using Corona mechanism". In: *9th IEEE International Symposium on Industrial Embedded Systems (SIES)*. 2014, pp. 200–209.

[Gna+09] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. "Collection Tree Protocol". In: *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. SenSys '09. Berkeley, California: ACM, 2009, pp. 1–14. URL: http://doi.acm.org/10.1145/1644038.1644040.

[Gon+11] A. Gonga, O. Landsiedel, and M. Johansson. "MobiSense: Power-efficient micro-mobility in wireless sensor networks". In: *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. 2011, pp. 1–8.

[Gun+08] S. Gundavelli, K. Leung, V. Devrapalli, K. Chowdhury, and B. Patil. *Proxy Mobile IPv6*. IETF RFC 5213. 2008.

[GV+09] L.J. García Villalba, A. L. Sandoval Orozco, A. Triviño Cabrera, and C. J. Barenco Abbas. "Routing Protocols in Wireless Sensor Networks". In: *Sensors* 9.11 (2009), p. 8399. URL: http://www.mdpi.com/1424-8220/9/11/8399.

[Ham+09] S.A. Hameed, E.M. Shaaban, H.M. Faheem, and M.S. Ghoniemy. "Mobility-aware MAC protocol for delay-sensitive wireless sensor networks". In: *International Conference on Ultra Modern Telecommunications Workshops (ICUMT)*. 2009, pp. 1–8.

[HC08] J. W. Hui and D. E. Culler. "IP is Dead, Long Live IP for Wireless Sensor Networks". In: *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*. SenSys '08. Raleigh, NC, USA: ACM, 2008, pp. 15–28. URL: http://doi.acm.org/10.1145/1460412.1460415.

[HP01] Z. H. Haas and M. R. Peralman. "The performance of query control schemes for the zone routing protocol". In: *IEEE/ACM Transactions on Networking* (2001).

[HT11] J. Hui and P. Thubert. *Compression Format for IPv6 Datagrams over IEEE 802.15.4-based Networks, IETF RFC 6282*. 2011.

[Hua+13a] P. Huang, L. Xiao, S. Soltani, M.W. Mutka, and N. Xi. "The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey". In: *IEEE Communications Surveys Tutorials* 15.1 (2013), pp. 101–120.

[Hua+13b] P. Huang, L. Xiao, S. Soltani, M.W. Mutka, and N. Xi. "The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey". In: *IEEE Communications Surveys Tutorials* 15.1 (2013), pp. 101–120.

[Hus+10] M.A. Hussain, N. Alam, S. Ullah, N. Ullah, and Kyung Sup Kwak. "TDMA based directional MAC for WBAN". In: *2010 6th International Conference on Networked Computing (INC)*. 2010, pp. 1–5.

[Inc+11] O.D. Incel, L. van Hoesel, P. Jansen, and P. Havinga. "MC-LMAC: A Multi-channel MAC Protocol for Wireless Sensor Networks". In: *Ad Hoc Networks* (Jan. 2011), pp. 73–94. URL: http://dx.doi.org/10.1016/j.adhoc.2010.05.003.

[Iot] *Seize New IoT Opportunities with the Cisco IoT System*. http://www.cisco.com/web/solutions/trends/iot/portfolio.html.

[Iov+13] O. Iova, F. Theoleyre, and T. Noel. "Stability and efficiency of RPL under realistic conditions in Wireless Sensor Networks". In: *IEEE 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*. 2013, pp. 2098–2102.

[Iov14] Oana Iova. "Standards Optimization and Network Lifetime Maximization for Wireless Sensor Networks in the Internet of Things". Theses. Universite de Strasbourg, Dec. 2014. URL: https://hal.archives-ouvertes.fr/tel-01113059.

[JK07] A. Jhumka and S. Kulkarni. "On the Design of Mobility-Tolerant TDMA-Based Media Access Control (MAC) Protocol for Mobile Sensor Networks". English. In: *Distributed Computing and Internet Technology*. Ed. by Tomasz Janowski and Hrushikesha Mohanty. Vol. 4882. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 42–53. URL: http://dx.doi.org/10.1007/978-3-540-77115-9_4.

[JM96] D. Johnson and D. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks". In: *Mobile Computing*. Vol. 353. The Kluwer International Series in Engineering and Computer Science. Springer US, 1996, pp. 153–181.

[Kar+12] P. Karkazis, H.C. Leligou, T. Orphanoudakis, and T. Zahariadis. "Geographical routing in wireless sensor networks". In: *2012 International Conference on Telecommunications and Multimedia (TEMU)*. 2012, pp. 19–24.

[Kas+11] M. Kassim, R.A. Rahman, and R. Mustapha. "Mobile ad hoc network (MANET) routing protocols comparison for wireless sensor network". In: *2011 IEEE International Conference on System Engineering and Technology (ICSET)*. 2011, pp. 148–152.

[Kim+07] K. Kim, S. Daniel Park, G. Montenegro, S. Yoo, and N. Kushalnagar. *6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD)*. IETF draft 03. 2007.

[KK00] Brad Karp and H. T. Kung. "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks". In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. MobiCom '00. Boston, Massachusetts, USA: ACM, 2000, pp. 243–254. URL: http://doi.acm.org.gate6.inist.fr/10.1145/345910.345953.

[KK08] J. Krnic and S. Krco. "Impact of WSN applications generated traffic on WCDMA access networks". In: *IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. 2008, pp. 1–5.

[Ko+14] J. Ko, J. Jeong, J. Park, J. Jun, N. Kim, and O. Gnawali. *RPL Routing Pathology In a Network With a Mix of Nodes Operating in Storing and Non-Storing Modes*. Internet-Draft draft-ko-roll-mix-network-pathology-04. IETF, 2014.

[Kor+12] I.E. Korbi, M. Ben Brahim, C. Adjih, and L.A. Saidane. "Mobility Enhanced RPL for Wireless Sensor Networks". In: *Proceedings of the 3rd International Conference on the Network of the Future (NOF)*. Tunis, Tunisia, 2012.

[KP06] R. S. Koodli and C. E. Perkins. "Mobility on the Internet: Introduction". In: *MOBILE Inter-Networking with IPv6*. John Wiley and Sons, Inc., 2006. URL: http://dx.doi.org/10.1002/9780470126486.ch1.

[Kun+13] R. Kuntz, J. Montavont, and T. Noel. "Improving the medium access in highly mobile Wireless Sensor Networks". English. In: *Telecommunication Systems* (2013).

[Kun10] Romain Kuntz. "Medium Access Control Facing the Dynamics of Wireless Sensor Networks". Theses. Université de Strasbourg, Sept. 2010. URL: https://tel.archives-ouvertes.fr/tel-00521389.

[KW10] D. Katz and D. Ward. *Bidirectional Forwarding Detection (BFD)*. IETF RFC 5880. 2010.

[Lee+09] S. H. Lee, S. Lee, H. Song, and H.S. Lee. "Wireless sensor network design for tactical military applications : Remote large-scale environments". In: *IEEE Military Communications Conference (MILCOM)*. 2009, pp. 1–7.

[Lee+12] K.C. Lee, R. Sudhaakar, L. Dai, S. Addepalli, and M. Gherla. "RPL Under Mobility". In: *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC)*. Las Vegas, NV, USA, 2012.

[Leg+08] J. Leguay, M. Lopez-Ramos, K. Jean-Marie, and V. Conan. "An efficient service oriented architecture for heterogeneous and dynamic wireless sensor networks". In: *IEEE International Workshop on Practial Issues in Building Sensor Network Applications (SensApp)*. 2008.

[Lev+04] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler. "The Emergence of Networking Abstractions and Techniques in TinyOS". In: *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*. NSDI'04. San Francisco, California: USENIX Association, 2004, pp. 1–1. URL: http://dl.acm.org/citation.cfm?id=1251175.1251176.

[Lev+05] P. Levis et al. "TinyOS: An Operating System for Sensor Networks". English. In: *Ambient Intelligence*. Ed. by Werner Weber, JanM. Rabaey, and Emile Aarts. Springer Berlin Heidelberg, 2005, pp. 115–148. URL: http://dx.doi.org/10.1007/3-540-27139-2_7.

[Lev+11] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. *The Trickle Algorithm*. IETF RFC 6206. 2011.

[Lew04] F. Lewis. "Wireless sensor networks". In: *Smart environments: technologies, protocols, and applications* (2004).

[Lor+04] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. "Sensor Networks for Emergency Response: Challenges and Opportunities". In: *IEEE Pervasive Computing* 3.4 (Oct. 2004), pp. 16–23. URL: http://dx.doi.org/10.1109/MPRV.2004.18.

[Mar+10] J. Martocci, P. De Mil, N. Riou, and W. Vermeylen. *Building Automation Routing Requirements in Low-Power and Lossy Networks*. RFC 5867. IETF, 2010.

[ME+08] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis. "Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks". In: *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*. IPSN '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 421–432. URL: http://dx.doi.org/10.1109/IPSN.2008.10.

[Mon+07] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. *Transmission of IPv6 Packets over IEEE 802.15.4 Networks*. IETF RFC 4944. 2007.

[Mon+14] J. Montavont, D. Roth, and T. Noel. "Mobile IPv6 in Internet of Things: Analysis, Experimentations and Optimizations". In: *Elsevier Ad Hoc Sensor Networks Journal* 14 (2014).

[Msp] *WSN430 Open Node*. `https://www.iot-lab.info/hardware/wsn430/`.

[Nar+07] T. Narten, E. Nordman, W. Simpson, and H. Soliman. *Neighbor Discovery for IP version 6 (IPv6)*. IETF RFC 4861. 2007.

[Nie+03] T. Nieberg, S. Dulman, P. Havinga, L. van Hoesel, and J. Wu. "Collaborative Algorithms for Communication in Wireless Sensor Networks". In: *Ambient Intelligence: Impact on Embedded System Design*. Boston: Kluwer Academic Publishers, 2003, pp. 271–294. URL: `http://doc.utwente.nl/65362/`.

[OMS09] M. Ordouei and M.T. Manzuri Shalmani. "A TDMA-Based MAC Protocol for Mobile Sensor Networks". In: *ICN '09. Eighth International Conference on Networks*. 2009, pp. 71–75.

[PAP08] R. de Paz Alberola and D. Pesch. "AvroraZ: Extending Avrora with an IEEE 802.15.4 Compliant Radio Chip Model". In: *Proceedings of the 3Nd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*. PM2HW2N '08. Vancouver, British Columbia, Canada: ACM, 2008, pp. 43–50. URL: `http://doi.acm.org/10.1145/1454630.1454637`.

[PC15] F. Peng and M. Cui. "An energy-efficient mobility-supporting MAC protocol in wireless sensor networks". In: *Journal of Communications and Networks* 17.2 (2015), pp. 203–209.

[Per+11] C. Perkins, D. Johnson, and J. Arkko. *Mobility support in IPv6*. IETF RFC 6275. 2011.

[Pet+06] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella. "Performance Study Of IEEE 802.15.4 Using Measurements and Simulations". In: *Proceedinngs of IEEE Wireless Communication and Networking Conference (WCNC)*. 2006.

[PJ04] H. Pham and S. Jha. "An adaptive mobility-aware MAC protocol for sensor networks (MS-MAC)". In: *2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems*. 2004, pp. 558–560.

[Pol+05] J. Polastre, R. Szewczyk, and D. Culler. "Telos: enabling ultra-low power wireless research". In: *International Symposium on Information Processing in Sensor Networks*. 2005.

[Raj+06] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves. "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks". English. In: *Wireless Networks* 12.1 (2006), pp. 63–78. URL: `http://dx.doi.org/10.1007/s11276-006-6151-z`.

[Rot+12] D. Roth, J. Montavont, and T. Noel. "Performance Evaluation of Mobile IPv6 over 6LoWPAN". In: *Proceedings of the 9th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*. PE-WASUN '12. Paphos, Cyprus: ACM, 2012, pp. 77–84. URL: `http://doi.acm.org.gate6.inist.fr/10.1145/2387027.2387041`.

[Sha+13] M. Sha, G. Hackmann, and C. Lu. "Energy-efficient Low Power Listening for wireless sensor networks in noisy environments". In: *2013 ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2013, pp. 277–288.

[She+12] Z. Shelby, S. Chakrabarti, E. Nordmark, and C. Bormann. *Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)*. IETF Request for Comments (RFC) 6775. 2012.

[Sik+06] P. Sikka, P. Corke, P. Valencia, D. Swain, and G. Bishop-Hurley. "Wireless ad hoc sensor and actuator networks on the farm". In: *ACM International Conference on Information Processing in Sensor Networks*. 2006.

[Sol+08] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier. *Hierarchical Mobile IPv6 (HMIPv6) Mobility Management*. IETF RFC 5380. 2008.

[St2] *M3 Open Node*. `https://www.iot-lab.info/hardware/m3/`.

[Sun+08] Y. Sun, O. Gurewitz, and D. B. Johnson. "RI-MAC: A Receiver-initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks". In: *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*. SenSys '08. Raleigh, NC, USA: ACM, 2008, pp. 1–14. URL: `http://doi.acm.org/10.1145/1460412.1460414`.

[Ter+08] F. Teraoka, K. Gogo, K. Mitsuya, R. Shibui, and K. Mitani. *Unified Layer 2 (L2) Abstractions for Layer 3 (L3)-Driven Fast Handover*. IETF RFC 5184. 2008.

[WA+06] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. "Deploying a wireless sensor network on an active volcano". In: *IEEE Internet Computing* 10.2 (2006), pp. 18–25.

[Wei+02] Y. Wei, J. Heidemann, and D. Estrin. "An energy-efficient MAC protocol for wireless sensor networks". In: *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. Vol. 3. 2002, 1567–1576 vol.3.

[Win+12] T. Winter et al. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*. RFC 6550. IETF, 2012.

[Wsn] *WSNet / Worldsens simulator*. `http://wsnet.gforge.inria.fr/`.

[Zan+07] E. Zanaj, M. Baldi, and F. Chiaraluce. "Efficiency of the gossip algorithm for wireless sensor networks". In: *15th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. 2007, pp. 1–5.

[ZD10] T. Zhiyong and W. Dargie. "A mobility-aware medium access control protocol for wireless sensor networks". In: *2010 IEEE GLOBECOM Workshops (GC Wkshps)*. 2010, pp. 109–114.

[Zho+06] G. Zhou, C. Huang, T. Yan, T. He, J.A. Stankovic, and T.F. Abdelzaher. "MMSN: Multi-Frequency Media Access Control for Wireless Sensor Networks". In: *25th IEEE International Conference on Computer Communications (INFOCOM)*. 2006, pp. 1–13.

[Zhu+13] W.T. Zhu, D.Y. Gao, H.K. Zhang, and W.C. Zhao. *Address Autoconfiguration for RPL*. IETF draft-zhu-roll-addr-autoconf-00. 2013.

[ZS09] Haibo Z. and Hong S. "Balancing Energy Consumption to Maximize Network Lifetime in Data-Gathering Sensor Networks". In: *IEEE Transactions on Parallel and Distributed Systems* 20.10 (2009), pp. 1526–1539.

[Bus14] Business Insider. *Here comes the Internet of Things*. Tech. rep. Business Insider Report, 2014.

[JP.12] JP. Vasseur et al. *Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks*. IETF RFC 6551. 2012.

List of Figures

List of Tables

List of Abbreviations

6CO	6LoWPAN Context Option.
6LBR	Border Router.
6LR	Router.
6LoWPAN	IPv6 over Low power WPAN.
ABRO	Authoritative Border Router Option.
ACK	Acknowledgment.
AODV	Ad hoc On-Demand Distance Vector.
ARO	Address Registration Option.
ARP	Address Resolution Protocol.
ARPANET	Advanced Research Projects Agency Network.
BE	Backoff exponent.
BFD	Bidirectional Forwarding Detection.
CAP	Contention Access Period.
CCA	Clear Channel Assessment.
CID	Context Identifier.
CoAP	Constrained Application Protocol.
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance.
CTP	Collection Tree Protocol.
CTS	Clear to Send.
DAG	Directed Acyclic Graph.
DAO	Destination Advertisement Object.
DARPA	Defense Advanced Research Projects Agency.
DIO	DODAG Information Object.
DIS	DODAG Information Solicitation.
DNS	Domain Name Server.
DODAG	Destination-Oriented Directed Acyclic Graph.
DSN	Distributed Sensor Networks.
DTSN	Destination Advertisement Trigger Sequence Number.
ETX	Expected Transmission Count.
EUI-64	64 bits Extended Unique Identifier.
FFD	Full Function Device.
GPS	Global Positioning System.

GPSR	Greedy Perimeter Stateless Routing.
GSP	Global Synchronization Period.
GTS	Guaranteed Time Slot.
HA	Home Agent.
HMIPv6	Hierarchical Mobile IPv6.
ICMPv6	Internet Control Message Protocol version 6.
IETF	Internet Engineering Task Force.
IP	Internet Protocol.
IPv4	Internet Protocol version 4.
IPv6	Internet Protocol version 6.
ISM	Industrial, Scientific and Medical radio bands.
LLN	Low-power and Lossy Network.
LOAD	6LoWPAN Ad Hoc On-Demand Distance Vector.
LOS	line of sight.
LPL	Low-Power Listening.
LPP	Low Power Probing.
LQI	Link Quality Indicator.
MAC	Medium Access Control.
MIPv6	Mobile IPv6.
MOP	Mode of Operation.
MSB	most significant bit.
MT-RPL	Mobility-Triggered RPL.
NA	Neighbor Advertisement.
ND	Neighbor Discovery.
NS	Neighbor Solicitation.
NUD	Neighbor Unreachability Detection.
O-QPSK	Offset-Quadrature Phase Shift Keying.
OF	Objective Function.
OSI	Open Systems Interconnection.
PDR	Packet Delivery Ratio.
PHR	physical header.
PIO	Prefix Information Option.
PMIPv6	Proxy Mobile IPv6.
RA	Router Advertisement.
ROLL	Routing Over Low power and Lossy networks.
RPL	IPv6 Routing Protocol for Low-power and Lossy Networks.
RS	Router Solicitation.
RSSI	Received Signal Strength Indicator.
RTS	Request to Send.
SHR	synchronization header.
SLLAO	Source Link-Layer Address Option.
SOSUS	Sound Surveillance System.
SYNC	SYNC.

TDMA	Time Division Multiple Access.
UDP	User Datagram Protocol.
WBAN	Wireless Body Area Network.
WSN	Wireless Sensor Network.