Medium Access Control Facing the Dynamics of Wireless Sensor Networks

Romain Kuntz

PH.D. THESIS

PRESENTED AT:
UNIVERSITY OF STRASBOURG
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
LSIIT (CNRS UMR 7005)

FOR OBTAINING THE DEGREE:
UNIVERSITY OF STRASBOURG
DOCTOR OF PHILOSOPHY (PH.D)
IN COMPUTER SCIENCE

MEDIUM ACCESS CONTROL FACING THE DYNAMICS OF WIRELESS SENSOR NETWORKS

by
ROMAIN KUNTZ

PUBLIC DEFENSE ON SEPTEMBER 17TH, 2010
WITH THE FOLLOWING JURY:

ERIC FLEURY, *External evaluator, Professor at ENS Lyon*
MARCELO DIAS DE AMORIM, *External evaluator, LIP6/CNRS Research Scientist*
THOMAS NOËL, *Thesis advisor, Professor at University of Strasbourg*
JULIEN MONTAVONT, *Co-advisor, Assistant Professor at University of Strasbourg*
DAVID SIMPLOT-RYL, *Examinator, Professor at LIFL Lille*
JEAN-JACQUES PANSIOT, *Examinator, Professor at University of Strasbourg*

REMERCIEMENTS

Une thèse n'est pas le fruit d'un seul individu, je profite donc de cet espace pour remercier l'ensemble des personnes qui m'ont suivi et soutenu ces trois dernières années.

Je tiens tout d'abord à remercier Thomas Noël (Professeur à l'Université de Strasbourg), mon directeur de thèse, pour m'avoir donné l'opportunité de démarrer une thèse à mon retour en France en 2007. C'est grâce à son encadrement, sa disponibilité, ses conseils et la liberté qu'il a su me donner que j'ai pu mener à bien ces recherches. La co-direction de ces travaux réalisée par Julien Montavont (Maître de Conférences à l'Université de Strasbourg) m'a continuellement permis d'en améliorer la qualité et la profondeur.

Je remercie Eric Fleury, Professeur à l'École Normale Supérieure de Lyon, et Marcelo Dias de Amorim, Chercheur CNRS au laboratoire LIP6 de l'Université Pierre et Marie Curie à Paris, d'avoir montré leur intérêt dans mes travaux en acceptant la charge de rapporteurs. Je remercie également David Simplot-Ryl, Professeur au Laboratoire d'Informatique Fondamentale de Lille, et Jean-Jacques Pansiot, Professeur à l'Université de Strasbourg, d'avoir bien voulu juger ces travaux en tant qu'examinateurs.

Cette aventure n'aurait toutefois peut-être jamais eu lieu si elle n'avait pas commencé par un stage au sein de l'équipe Réseaux et Protocoles lors de l'été 2003. Il est donc tout naturel que j'exprime à nouveau ma gratitude envers Jean-Jacques Pansiot pour m'avoir accepté au sein de son équipe, et qui depuis a toujours fait preuve d'une grande disponibilité et d'une justesse remarquable.

Les trois années qui séparent l'obtention de mon diplôme de DESS et le début de la thèse furent marquées par un voyage qui m'a définitivement convaincu de m'orienter vers la recherche. Je remercie Thierry Ernst pour m'avoir initié à mes premiers travaux lors de mon séjour à l'Université de Keio au Japon. Le soutien de Koshiro, Martin, Guillaume, Sergio, Ryuji, du *Master* et d'un bon nombre de membres du projet WIDE s'est aussi révélé extrêmement précieux.

Je voudrais également remercier tous les membres de l'équipe Réseaux et Protocoles, et plus particulièrement Antoine Gallais et Julien Montavont, pour les innombrables discussions devant la machine à café, qui furent certainement à l'origine de l'ensemble des contributions présentes dans ce manuscrit. Un grand merci à Guillaume Schreiner, Erkan Valentin, Sebastien Vincent et Nicolas Weber pour leurs travaux sur les capteurs. L'ambiance environnante menée par les anciens, actuels et futurs doctorants (Emil,

Alex, Vincent L., Damien, Julien B.) a aussi largement contribué à l'achèvement de ces travaux.

Evidemment, une pensée profonde va à l'ensemble de ma famille, et tout particulièrement à mes parents, ma grand-mère et mes soeurs (et Geoffroy !), qui m'ont toujours soutenu et qui continueront sans aucun doute à le faire, proche ou à distance. Merci pour les tup', les week-ends au vert et tant d'autres choses !

Mes amis et amies ont également toute ma gratitude pour les nombreuses soirées qui m'ont permis de passer trois superbes années à Strasbourg. La liste est longue, ils et elles se reconnaîtront.

ABSTRACT

A Wireless Sensor Network (WSN) consists in spatially distributed autonomous and embedded devices that cooperatively monitor physical or environmental conditions in a less intrusive fashion. The data collected by each sensor node (such as temperature, vibrations, sounds, movements etc.) are reported to a sink station in a hop-by-hop fashion using wireless transmissions. In the last decade, the challenges raised by WSNs have naturally attracted the interest of the research community. Especially, significant improvements to the communication stack of the sensor node have been proposed in order to tackle the energy, computation and memory constraints induced by the use of embedded devices. A number of successful deployments already denotes the growing interest in this technology.

Recent advances in embedded systems and communication protocols have stimulated the elaboration of more complex use cases. They target dense and dynamic networks with the use of mobile sensors or multiple data collection schemes. For example, mobility in WSNs can be employed to extend the network coverage and connectivity, as well as improve the routing performances. However, these new scenarios raise novel challenges when designing communication protocols.

The work presented in this thesis focuses on the issues raised at the Medium Access Control (MAC) layer when confronted to dynamic WSNs. We have first studied the impact of mobility and defined two new MAC protocols (*Machiavel* and *X-Machiavel*) which improve the medium access of mobile sensor nodes in dense networks. Our second contribution is an auto-adaptive algorithm for preamble sampling protocols. It aims at minimizing the global energy consumption in networks with antagonist traffic patterns by obtaining an optimal configuration on each node. This mechanism is especially energy-efficient during burst transmissions that could occur in such dynamic networks.

RÉSUMÉ

Un réseau de capteurs sans fil (*Wireless Sensor Network*, WSN) consiste en une distribution spatiale d'équipements embarqués autonomes, qui coopèrent de manière à surveiller l'environnement de manière non-intrusive. Les données collectées par chaque capteur (tels que la température, des vibrations, des sons, des mouvements etc.) sont remontées de proche en proche vers un puits de collecte en utilisant des technologies de communication sans fil. Voilà une décennie que les contraintes inhérentes à ces réseaux attirent l'attention de la communauté scientifique. Ainsi, de nombreuses améliorations à différents niveaux de la pile de communication ont été proposées afin de relever les défis en termes d'économie d'énergie, de capacité de calcul et de contrainte mémoire imposés par l'utilisation d'équipements embarqués. Plusieurs déploiements couronnés de succès démontrent l'intérêt grandissant pour cette technologie.

Les récentes avancées en termes d'intégration d'équipements et de protocoles de communication ont permis d'élaborer de nouveaux scénarios plus complexes. Ils mettent en scène des réseaux denses et dynamiques par l'utilisation de capteurs mobiles ou de différentes méthodes de collection de données. Par exemple, l'intérêt de la mobilité dans les WSNs est multiple dans la mesure où les capteurs mobiles peuvent notamment permettre d'étendre la couverture d'un réseau, d'améliorer ses performances de routage ou sa connexité globale. Toutefois, ces scénarios apportent de nouveaux défis dans la conception de protocoles de communication.

Ces travaux de thèse s'intéressent donc à la problématique de la dynamique des WSNs, et plus particulièrement à ce que cela implique au niveau du contrôle de l'accès au médium (*Medium Access Control*, MAC). Nous avons tout d'abord étudié l'impact de la mobilité et défini deux nouvelles méthodes d'accès au médium (*Machiavel* et *X-Machiavel*) qui permettent d'améliorer les conditions d'accès au canal pour les capteurs mobiles dans les réseaux denses. Notre deuxième contribution est un algorithme d'auto-adaptation destiné aux protocoles par échantillonnage. Il vise à minimiser la consommation énergétique globale dans les réseaux caractérisés par des modèles de trafic antagonistes, en obtenant une configuration optimale sur chaque capteur. Ce mécanisme est particulièrement efficace en énergie pendant les transmissions par rafales qui peuvent survenir dans de tels réseaux dynamiques.

CONTENTS

GENERAL INTRODUCTION

The purpose of the work presented in this dissertation is to optimize the Medium Access Control (MAC) in dynamic Wireless Sensor Networks (WSNs). We have particularly focused on improving the integration of mobile sensors in large-scale, unattended networks, as well as enhancing the overall performances of networks in which burst transmissions occur. The contributions of this thesis are the definition of two new MAC protocols for mobile sensor nodes and one algorithm to make these protocols auto-adaptive to the traffic load.

WIRELESS SENSOR NETWORKS

A WSN consists in spatially distributed autonomous and embedded devices that cooperatively monitor physical or environmental conditions. The data collected by each node (such as temperature, vibrations, sounds, movements etc.) are reported to a *sink* station in a hop-by-hop fashion using wireless transmissions. Such data can then be processed and analyzed for a better understanding of the monitored environment. The need to perform these operations in a least intrusive fashion has motivated the development of small and low-power wireless devices. We will further detail hereinafter the main characteristics and constraints of such embedded systems. A number of successful deployments already denotes the growing interest in this technology. We will also overview the main applications of WSNs nowadays, and what possible shortcomings would prevent the envisioned use cases to be deployed.

Main characteristics

Thanks to the miniaturization of electronic devices and advances in wireless telecommunication, it became possible to build small communicating embedded devices at low costs. Coupled with physical sensors (such as temperature, sound, motion or pressure sensors), they can retrieve and report information from the surrounding environment. These small devices, later referred to as *sensor nodes* (Figure 1), dispose of limited hardware and resources as detailed in Table 1.

The main hardware components of a sensor node are the MicroController Unit (MCU), one or multiple physical sensors, a wireless transmitter and a battery. The MCU is a low-power platform which provides low processing and storage capabilities compared to nowadays computers, Personal Digital Assistants (PDAs) or even cellphones. The sensor measures a physical quantity and translates it into a digital signal. Such data may be processed by the MCU before being sent using the low-power wireless transmitter, which can communicate up to a distance of tens or hundreds

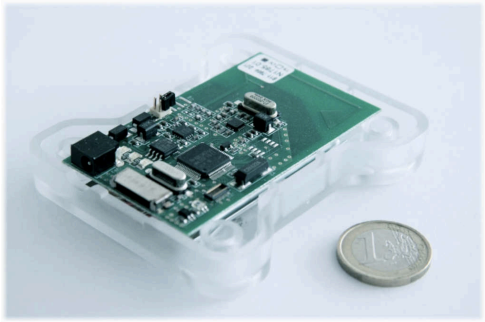


Figure 1: An example of wireless sensor node: the WSN430 [87]

of meters (according to the transmission power and frequency range used). In order to obtain a complete autonomous and wireless system, a sensor node is powered by a battery. Nowadays, sensor nodes are usually built from the example hardware detailed in Table 1. For instance, the TelosB mote [24] embeds the TI MSP430 MCU with a Chipcon CC2420 wireless transmitter and can be powered with two AA cells. The WSN430 sensor node [87] depicted in Figure 1 is also built from a TI MSP430 MCU, a Chipcon CC1100 wireless transmitter and is equipped with a PoLiFlex battery. On the software side, multiple embedded operating systems have been specifically designed for sensor nodes. Among others, TinyOS and Contiki provide the drivers and Application Programming Interfaces (APIs) to build applications on most of the existing sensor platforms.

Component	*Example*	*Characteristics*
MCU	TI MSP430 [96]	16-bit RISC CPU (8 - 25 Mhz) 128 B to 16 KB RAM 0.5 KB to 256 KB Flash
	Atmel ATmega 128 [10]	8-bit RISC CPU (16 Mhz) 4 KB RAM, 128 KB Flash
Physical sensor	Sensirion SHT11 [86]	Humidity and temperature
	Hamamatsu S1087 [40]	Visible light
Wireless transmitter	Chipcon CC1100 [94]	315/433/868/915 MHz up to 250 Kbps
	Chipcon CC2420 [95]	2400 - 2483.5 MHz up to 250 Kbps
Battery	AA Alkaline cell	1700 - 3000 mAh, 1.5 V
	AA NiMH cell	1300 - 2900 mAh, 1.2 V
	VARTA PoLiFlex [104]	830 mAh, 3.7 V, 2.2 mm slim
Operating system	TinyOS [25]	Component-based architecture
	Contiki [28]	Small footprint

Table 1: Main components of a sensor node.

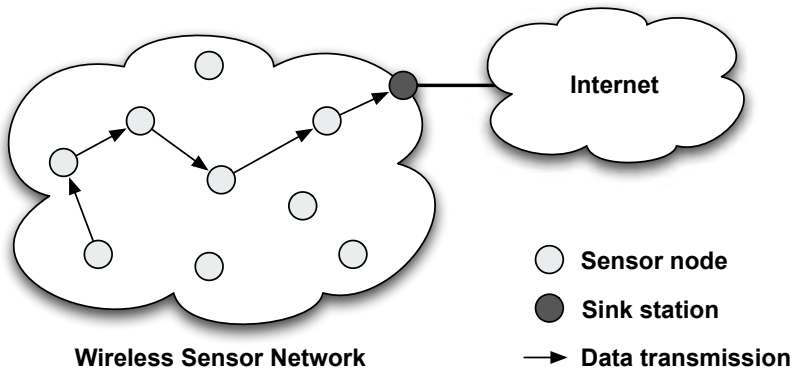


Figure 2: Overview of a Wireless Sensor Network. Data collected by a sensor node is reported toward a sink station in a hop-by-hop fashion.

The small size and weight, the low cost of the hardware and the ease of deployment of such platforms enable the sensing of the environment in a least intrusive fashion. By spatially distributing tens or hundreds of such autonomous devices, a Wireless Sensor Network (WSN) can be built in order to cooperatively monitor physical or environmental conditions at different locations. The low-power wireless transmitter usually mandates the collected data to be sent over multiple hops toward one or several collecting entities called a *sink* (Figure 2). Also, packet delivery performances are improved at low-power transmissions [116] which stimulates the use of multiple short hops rather than a single one over a long range link. The sink interconnects the WSN with a standard network (e.g. a private network, or the Internet) from which the WSN can be remotely accessed and monitored. The way the information is collected in the WSN depends on the application scenario, as we will detail below.

Application overview through existing deployments

The pervasive and ubiquitous aspects of WSNs have brought them in the front scene in the last decade. Indeed, sensing and reporting information in a transparent manner is one of their main advantages. This has naturally led to a number of WSNs deployed for environmental, habitat or structural monitoring purposes, as well as for surveillance systems or home automation. An overview of the characteristics of these deployments can give us an idea of the variety of applications offered by the WSNs.

The nature of the data traffic reveals two major and distinct categories of applications. They are summarized in Table 2. The first one gathers deployments that aim at continuously monitoring a phenomenon: habitat [67, 93, 115] and environment [14, 41, 58, 99] monitoring belong to this category. These deployments intend to

Cat.	*Use case*	*Data collection scheme*	*Specific constraints*
1	Habitat monitoring	Time-driven	Non-intrusive, hardly accessible, energy
	Environment monitoring	Time-driven	Sturdiness, energy
2	Structural health monitoring	Event-driven/ Query-driven	High frequency sampling, energy
	Surveillance system	Event-driven/ Query-driven	Security, energy
1+2	Home automation	Event-driven/ Time-driven	Security, Heterogeneity

Table 2: Main use cases of WSNs nowadays and their constraints.

better understand a situation by periodically collecting samples over a long time interval: the sensor nodes report their measured values toward a sink in a *time-driven* fashion. In order to achieve low-power operations, this data collection scheme usually mandates the nodes to operate with a small sampling rate and low data rates. The collected data is processed once a significant amount of information has been gathered, and its analysis often helps scientists to approach the phenomenon from a new point of view. For example, an analysis of the reported data in the SensorScope glacier monitoring system [14] enabled the modeling of a particular micro-climate, thus allowing flood monitoring and prediction.

The second category covers deployments that measure a response to stimuli. This includes for example structural health monitoring [19, 49] as well as surveillance or intrusion detection systems [42, 62]. The main purpose is to collect as much information as possible only upon specific events. A *query-driven* or *event-driven* data collection scheme is more suitable for this category of applications than a time-driven one. In the query-driven mode, the sink initiates the data collection by explicitly sending a request to one or more sensor nodes. In the event-driven mode, the sensor nodes report their measured values toward the sink when they reach a certain threshold. Sometimes both schemes are combined [106], where the sink starts collecting data upon reception of various messages triggered by an interesting event on the sensor nodes. These use cases usually require high data rates, high fidelity sampling (through a reliable end-to-end protocol), precise time-stamping and hence efficient time synchronization. The collected data may either be processed upon reception (e.g. in a surveillance system) or stored for a later analysis. As an example, monitoring the vibrations due to the adjacent road traffic

of the Torre Aquila monument [19] helped scientists to reproduce the structural behavior of the building.

Both categories are sometimes combined in order to monitor long term and sporadic events at the same time. For example, home automation systems such as described in [38] can be deployed to continuously trace the indoor temperature of a building as well as fire an alarm only when smoke is detected.

As a side note, we can emphasize that use cases and characteristics of Wireless Sensor Networks distinguish them from both ad hoc networks and Delay-Tolerant Networks (DTNs).

Common characteristics in existing WSNs

A thorough analysis of the WSNs deployed so far exhibits several characteristics common to both categories previously described. They are summarized in Table 3.

In terms of size, only relatively small-scale networks have been deployed, of the order of tens of sensors usually divided to smaller patches. The topology is always carefully chosen and the radio power tuned to guarantee the desired connectivity from startup. The nature of the data traffic is convergecast, the information being conveyed from n sensor nodes to one or a few sinks. Most of the deployments are single-hop, sometimes a few hops are considered but hardly up to 6 hops. Although one exception has achieved 46 hops [49], it used a linear topology which significantly eased the routing and link management.

The length of deployments is usually less than a year, with most of WSNs being deployed from just a few days to a few weeks. This makes the energy management easier on the sensor nodes. Even

Characteristics	*Today's deployments*	*Impact on the design*
Size of the network	Relatively small (tens of nodes)	Small collision domain
Nature of the traffic	Convergecast (n-to-1)	Simplified routing scheme
Topology	Single to few hops	Simplified link management
Sensors placement	Carefully studied	Simplified neighborhood management
Length of deployment	Less than a year	Energy management is a secondary concern
Mobility	Limited to a few number of nodes	Simplified scheduling scheme

Table 3: Common characteristics of WSN deployments nowadays.

though energy-efficiency has been a concern in deployments, cells in addition to solar panels are usually enough to sustain the energy demand during the experiment [14].

The interest in deployments of mobile WSNs has recently increased [6,34,98,115], and mobility is foreseen as a likely solution to expand the network coverage by reaching areas that need to be monitored [114], improve the routing performances or the overall connectivity by replacing failed routing nodes [26]. Sensor nodes may also move from their locations because of the environment in which they evolve (such as water or air). Of course, mobile sensors bring new challenges (e.g. in terms of scheduling or routing) but their induced constraints remained accessible so far. In [6], a total of 5 mobile sensors has been employed. Data was reported using periodic flooding to a sink located hardly ever more than 2 hops away. Even though the experiment presented in [98] had hundreds of mobile devices, they were actually grouped into four patches in which single-hop communications occurred only few times a day. In every mobile scenario, the small number of nodes or the low frequency of communications only increased slightly the complexity of the deployments.

As the current deployment characteristics result from basic constraints, scientists could design protocols with simple features while still matching the target application. For instance, the people in charge of the SensorScope deployment [14] equipped their sensing stations with solar cells. The evaluation of the daily energy contribution allowed them to design a MAC layer without necessarily focusing on complex energy-saving mechanisms. Specifically, they came up with a high radio utilization scheme (all sensor nodes being simultaneously active during 12 seconds every 2 minutes) that still ensured a successful long term experiment. In [67], the static and linear aspect of the network topology has led the authors to opt for a novel time-division scheme to control the medium access. In fact, this method performed well provided that the network topology does not change.

Hence, we could record a significant number of existing deployments relying on protocols built from scratch (e.g. in [14,42,67,115]). The constraints induced by the application and the chosen hardware indeed governed a specific design of the MAC layer. As already mentioned, it resulted in successful deployments but it also contributed to making the reuse of these solutions difficult. In turn, these potential difficulties led to more and more MAC layers built from scratch. This vicious circle raises two major drawbacks. First, it undoubtedly imposes WSN deployments to be performed by networking experts with strong programming skills in embedded systems. Second, the implemented solutions may become barely usable once confronted to slightly different deployment constraints.

Envisioned applications and shortcomings

With time, these deployment constraints will obviously raise new challenges when designing a suitable communication stack for the sensor nodes. Each of the characteristics presented in Table 3 will be applied on a broader scale: increase in the number of nodes, multi-hop topology, longer deployment, etc. Foreseen scenarios on the use of WSNs for natural disasters expect to deploy the sensors from airplanes, resulting in a random topology at large scale in harsh environments [31]. In such case, the difficulty to physically access the devices (e.g. to replace the battery) is balanced with redundancy, hence a higher density of nodes. Due to re-deployment or death of some sensors, as well as the instability of the wireless links, the topology of the network and the way it transports the information would evolve during the lifetime of the deployment. In parallel, further miniaturizations of sensor nodes will make Body Area Networks (BANs) a reality [13]. By measuring the blood pressure or the heart rate, patients will be remotely monitored while on the move, in a least intrusive fashion. Mobility has a leading role in such medical applications, as depicted in [89].

In light of this, most of the protocols employed in today's deployments would certainly face scalability issues if the same experiment was to be performed at a larger scale. First, they have not been designed to operate in large and dense networks. Due to the convergecast traffic pattern, nodes located around the sink would need to handle much more traffic than the others. The static aspect of nowadays deployments makes them also very unlikely to behave efficiently in uncharted environments or where mobility would have a major role. As the collected data is not analyzed in real-time, short delays have not been considered yet. This may not be acceptable for responsive applications, such as target detection and tracking, that have rarely been deployed so far. The rather short length of deployments or the use of solar panels does not give a great importance to power management. This has certainly kept engineers from optimizing the communication stack in terms of energy consumption. For example, the 10% radio usage of the SensorScope system could be considered as excessive in WSNs [30]. Instead, sensor nodes are expected to operated unattended for long periods of time.

The scientists community is already acquainted with these aspects: research on algorithms and communication protocols in WSNs has yielded a tremendous effort in the last decade. Communication protocols (at the transport, routing and MAC layers) have been the field of numerous improvements [112], especially to benefit from several services such as localization, coverage, synchronization or security. The common point of these

researches is the systematic attempt to design energy-efficient solutions [7]. Although substantial efforts have already been achieved on the hardware components of a sensor node (such as gains in the battery capacity and recharge systems [20], and low-power micro-controllers [10, 96]), the communication device remains the most consuming component of the node. For example, the TI MSP430 micro-controller consumes 0.2 mA in active mode (at 1 MHz, 2.2 V [96]) whereas the CC1100 wireless transmitter drains 16.9 mA in transmit mode (at 0 dBm, 868/915 MHz [94]) and 16.4 mA in receive mode (at 250 kbps). A radio constantly transmitting or receiving would empty one AA cell (2500 mAh) in about 6 days. This is the reason why the MAC layer has been the primary source of optimizations. Indeed, the MAC can control the radio utilization by periodically turning it off (to save energy) and on (to participate in network communications), hence taking part in more energy conservation.

Still, this dynamic side of future deployments would require such MAC protocols to perform well in various situations. Mobility has been considered for a while as one of the future issues that need to be addressed at the MAC layer [4]. Changes in the network topology as previously described, or in the data collection scheme while the network is operating would imply time-varying or spatially non-uniform traffic loads. This excludes any prior installation of a monolithic MAC layer on the nodes. Instead, the way to access the medium should be adjusted according to the constraints of the moment.

THESIS ORGANIZATION

The work presented in this thesis focuses on the medium access in dynamic WSNs. As already discussed, the dynamics of a WSN can be attributable to a change in its topology, being physical (sensor nodes appearing or disappearing in the network due to their death, re-deployment or movements), or overlaid (variation of the neighborhood due to the instability of the wireless links). Such network dynamics can also occur from a change in the way the information is collected in the network (for example sensor nodes switching from an event-driven scheme to a time-driven one upon an event).

Do the existing proposals address these scenarios efficiently? Are they suitable when large number of nodes operate in the network? This thesis aims at answering these questions as well as tackling the issues of two specific aspects of dynamic WSNs at the MAC layer: the use of mobile sensor nodes and the change of the data collection scheme operated in the network.

Research context

The algorithms developed to access the medium in WSNs differ from the ones usually employed in traditional wireless or ad hoc networks. They focus on minimizing the radio usage while avoiding deafness and maintaining the layer-2 connectivity among the sensor nodes. Chapter 1 will present the main causes of energy depletion at the MAC layer and the major contributions toward energy-efficiency. Among them, we will especially focus in Chapter 2 on the ones that concentrate on mobile scenarios and that propose versatile features in order to adapt to the traffic conditions. We will particularly analyze their limits when confronted to the challenges existing in forthcoming deployments.

Accessing the medium in highly mobile WSNs

Mobile sensor nodes can experience several issues when accessing the medium, such as the difficult integration in the neighborhood schedule. So far, MAC protocols designed for mobile sensors did not focus on dense topologies nor scenarios where the mobile sensor nodes numerically exceed the fixed ones. Chapter 3 will exhibit the issues that can arise when a mobile sensor node roams across the network. Our first contribution, *Machiavel*, is presented in Chapter 4 and aims at solving these issues. The *X-Machiavel* protocol, detailed in Chapter 5, enhances this work by considering a high ratio of mobile sensor nodes traveling in the network.

From versatility to auto-adaptation

Versatile protocols allow the pre-configuration of the MAC layer prior to the deployment. However, adjustments while the network is operating can be hazardous as a reconfigured sensor node may not be able to communicate with its neighbors anymore. Such operations may however be necessary to ensure efficient operations in networks where the data collection scheme evolves with time, provoking burst transmissions. In Chapter 6, we will evaluate the performances of two well-reputed MAC protocols in such circumstances. From this analysis, we will present in Chapter 7 our second contribution, an algorithm which aims at making these protocols auto-adaptive.

In the conclusion, we will outline the main contributions of this thesis and present the possible perspectives of this work. In the Appendix A, we will summarize other works conducted in the Internet Protocol version 6 (IPv6) world. Appendix B is a summary of this thesis in French.

Part I

RESEARCH CONTEXT

1

MEDIUM ACCESS IN WIRELESS SENSOR NETWORKS

Contents

This Chapter introduces the functions assumed by the Medium Access Control (MAC) layer, the main design issues when operated on a sensor node, and the major schemes that were proposed so far to leverage these issues. A tremendous amount of work has been conducted in the last decade in this area (a recent survey [12] registers more than 70 MAC protocols especially designed for WSNs). We will therefore concentrate on seminal contributions that target energy-efficiency.

Most of the protocols detailed in the literature are built on these pioneering schemes and propose enhancements intended for specific scenarios. The ones that focus on network dynamics will be detailed in Chapter 2.

1.1 DEFINITION

The MAC is a sub-layer in the data link layer (layer 2) of the Open System Interconnection (OSI) model. Located below the Logical Link Control (LLC) sub-layer (which provides multiplexing and flow control mechanisms), the MAC is in charge of coordinating the access to the medium shared by several nodes. Especially, it dictates when a node can transmit or must listen to the channel in a way that fairness, reliability, scalability, low latency and fair throughput are guaranteed. This is all the more challenging in wireless environments as the medium is half-duplex and broadcast by nature: nodes cannot transmit and receive at the same time, and all the nodes located in the radio neighborhood of the transmitter overhear the data [107]. Coordination is thus important to avoid *collisions* (simultaneous transmissions on the

medium, which result in a jammed signal) or *deafness* (when the recipients are not ready to receive, e.g. if their radio is in transmission or sleeping mode). For that purpose, accessing the medium in wireless networks can be achieved using circuit or packet mode, as detailed below.

1.1.1 *Circuit mode MAC*

In circuit mode, a dedicated channel is established between nodes before the communication starts. Each dedicated circuit cannot be used by other nodes until it has been released. This mode offers a constant bandwidth, but the number of nodes simultaneously sharing the channel is limited. Among the circuit mode MAC, the following methods have emerged (Figure 3):

- *Time Division Multiple Access (TDMA)* divides the signal into different time slots that are distributed to each pair of nodes;
- *Frequency Division Multiple Access (FDMA)* allocates peers with different carrier frequencies of the radio spectrum;
- *Code Division Multiple Access (CDMA)* defines different code sequences that are assigned to each pair of nodes. Multiple transmitters can be multiplexed over the same channel.

All of these circuit mode MAC prevent collisions from occurring. However, CDMA-based schemes require additional circuitry and computations, as well as a large bandwidth to accommodate multiple simultaneous transmitters (the modulated and multiplexed signal has a much higher bandwidth that the communicated data). These constraints prevent them from being used in low-cost embedded systems. Sometimes used in combination, TDMA and FDMA schemes are dependent on the topology of the network, and respectively require time slot and frequency allocation algorithms, which is complex to achieve in large, multi-hop networks. TDMA schemes also require time synchronization methods to guarantee a common schedule among the nodes.

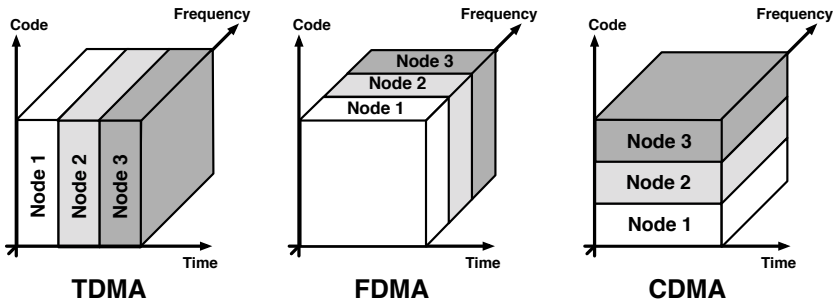


Figure 3: Overview of the TDMA, FDMA and CDMA schemes.

1.1.2 *Packet mode MAC*

In packet mode, each packet is individually addressed. No dedicated circuit or channel needs to be established. Hence, the number of nodes that can communicate over the same channel is not limited in theory. The access to the medium is contention-based, meaning that the nodes have to compete to transmit their data. Packet mode MAC protocols thus provide only best effort services, i.e. variable bandwidth and delay.

In succession ALOHA [1], packet mode schemes are nowadays based on *Carrier Sense Multiple Access (CSMA)*, where nodes verify the absence of other traffic on the medium by performing a channel sampling after waiting a random time (the *backoff*) and before transmitting. However, the fact that no dedicated channel is established cannot prevent frame collisions from occurring, especially in wireless environments, as depicted in Figure 4. Various CSMA schemes have thus emerged to deal with collisions. We can particularly mention the Collision Avoidance (CA) variant which is employed in the IEEE 802.11 Distributed Coordination Function (DCF) standard [46] for traditional wireless networks. With CSMA/CA, a node informs all others of its intent to transmit by using a Request To Send (RTS) message. The reception of a Clear To Send (CTS) from its peer implies that the channel has been reserved during the time of the communication, which triggers the transmission of the data frame. This scheme is particularly efficient to inhibit the *hidden node* problem that exists in wireless networks [81], but does not completely eliminate frame collisions.

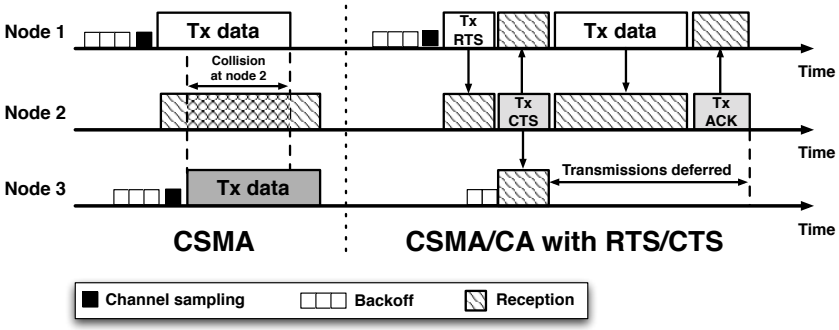


Figure 4: Overview of the CSMA scheme. Node 3 is not within the radio range of node 1, and thus does not hear its transmissions: collisions may happen at node 2. The use of RTS/CTS messages can mitigate this issue.

Research on MAC protocols for WSNs has particularly focused on solving the above-mentioned issues of packet and circuit mode MAC while keeping in mind energy efficiency. Before reviewing the main contributions in this area, we propose to detail the main causes of energy depletion in wireless communications.

1.2 MAJOR CAUSES FOR ENERGY DEPLETION

Idle listening, control packet overhead, overhearing and collisions have been identified as the main sources of energy wastage in WSNs [112]:

- *Idle listening* occurs when a node keeps listening to the medium while waiting for a frame (① in Figure 5). As nodes are not supposed to know when they will receive a message, their radio must be kept in receiving mode to avoid missing frames. As already mentioned, reception drains a lot of energy (16.4 mA for a CC1100 radio at 868/915 MHz [94]) even when no data is transiting, hence making idle listening the most significant source of energy wastage. This is all the more true under low traffic conditions where the channel is idle most of the time. Instead, the MAC should switch the radio off as much as possible while avoiding deafness, as performed for example by IEEE 802.11 Power Saving Mode (PSM) [46]. PSM is however not designed for multi-hop networks;

- *Control packets* are protocol overheads as they do not carry any useful information for the application layer but still consume energy for their transmission and reception (② in Figure 5). For example, the RTS/CTS messages used in IEEE 802.11 DCF are considered as high overheads in WSNs because their size is similar to the one of typical sensor data packets;

- *Overhearing* happens when a node receives a redundant broadcast packet, or a unicast packet that was destined to another neighbor (③ in Figure 5). The wireless medium being broadcast by nature, all the nodes located in the radio

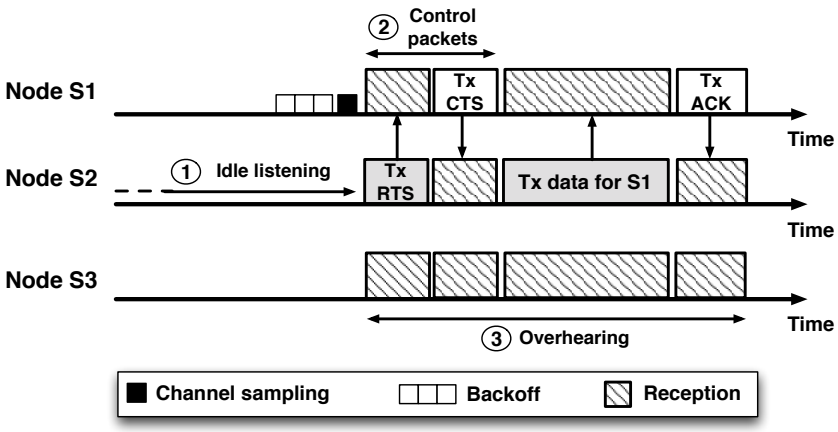


Figure 5: Idle listening, control packets and overhearing are some of the causes for energy depletion in WSNs.

range of the transmitter receives the frame, which will be discarded only after examination of its header. Because such reception uselessly drains energy, overhearing can particularly be a problem in dense deployments;

- *Collisions* occur when a node is within the transmission range of two or more nodes that are simultaneously emitting, as depicted previously in Figure 4. It may happen in the *hidden node* configuration for example [81]. In that case, the receiver cannot capture any frame. The energy consumed during the transmission and reception of these frames is wasted, and additional energy is required for the retransmission. As a result, the additional traffic reduces the channel availability which could cause even more collisions. As performances can seriously be degraded, CSMA-based MAC protocols should try to reduce them.

In light of this, it becomes obvious that accessing the medium in WSNs requires specific protocols that would mitigate the above-mentioned sources of energy dissipation.

1.3 MAIN CONTRIBUTIONS TOWARD ENERGY-EFFICIENCY

Among the set of functionalities provided by a MAC layer, scheduling has been the field of several enhancements in order to achieve drastic energy savings. The main idea is to put the radio in the sleep mode as often as possible while avoiding deafness and ensuring the link connectivity among the sensor nodes. Protocols aim at obtaining the smallest *duty cycle* (the duration of the radio active period, expressed in percentage) as possible, which gives an idea of the energy-efficiency of the protocol through the radio utilization. In order to meet this requirement, two main schemes initially came up: the synchronized protocols and the preamble sampling protocols. More recently, hybrid protocols have also emerged to combine the benefits of both schemes [12].

1.3.1 *Synchronized protocols*

Synchronized protocols organize the nodes around a common schedule and hence rely on time synchronization. The level of synchronization however differs according to the protocol: the slotted schemes require a tight synchronization while the schemes relying on a common active/sleep period are less restrictive on that matter. We detail both of them below.

Slotted schemes

Slotted schemes are based on the TDMA principles. Time is divided into slots distributed among the nodes, which agree with one another to use such slots to send or receive data, or to power off the radio. This scheme guarantees a collision-free slot to the sensor nodes. It is therefore particularly suitable to handle periodic traffic.

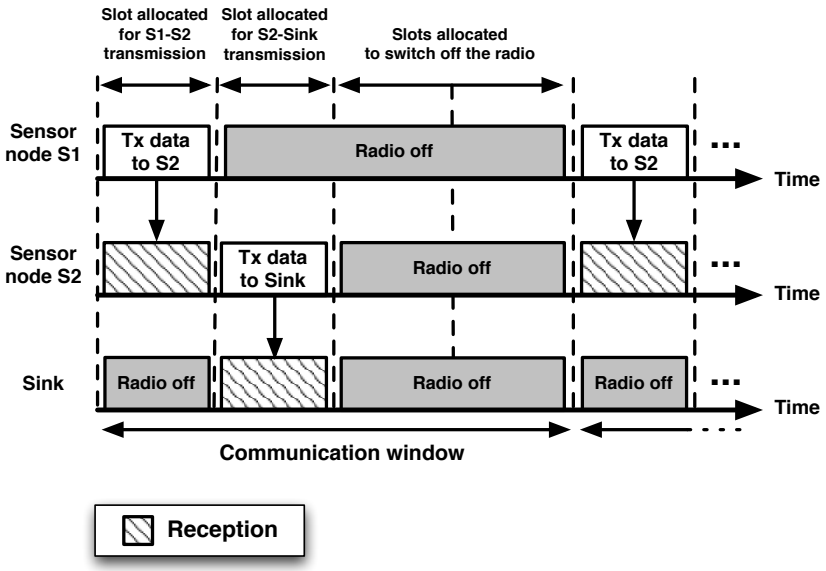


Figure 6: An example of slotted protocol with 3 sensor nodes sharing a communication window composed of 4 time slots.

A simple example is depicted in Figure 6. It displays a TDMA scheme which divides the communication window into 4 time slots. These slots are distributed among 3 sensor nodes (*S1*, *S2* and a sink station) that are placed in a linear topology where S1 is the origin of the data. In this example, one slot has been assigned for the communication from S1 to S2, another one for the transmission from S2 to the sink, and the 2 other slots are dedicated to switch off the radio of all the sensor nodes. In addition, when a node is not involved in a time slot, it can also put its radio into sleep mode. In the example depicted in Figure 6, we can point that S1 and the sink have a 25% duty cycle (1 active slot over the 4 slots of the communication window), and S2 has a 50% duty cycle (2 active slots in the communication window). This communication window is duplicated over time as long as the network operates.

From this example, we can perceive that the network topology, the desired bandwidth and the intended duty cycle have a strong influence on how the communication window will be organized. The number of slots increases with the number of sensor nodes that wish to participate in the communication. Targeting a low duty cycle also implies the allocation of dedicated slots to switch

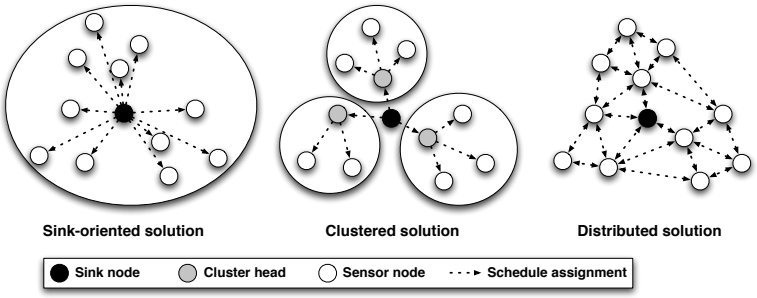


Figure 7: Sink-oriented, clustered or distributed schedule assignment.

off the radio. This mandates a longer communication window, hence longer delays and a lower available bandwidth. Spatial re-use of the time slots and frequencies (when used in combination with FDMA schemes) are thus often considered to avoid a too large communication window.

In terms of energy savings, slotted protocols reduce overhearing and idle listening: a sensor node switches off its radio when it is neither a transmitter nor a receiver in a time slot, and wakes it up only during its dedicated time slot. The collision-free schedule also mitigates retransmissions. However, the establishment and maintenance of such schedule requires control messages among the sensor nodes, which represent a non-negligible protocol overhead. Algorithms in the literature focus on how time slots (and sometimes frequencies) can be distributed among the sensor nodes in the network and maintained over time at a least possible cost.

For that purpose, multiple approaches have been suggested as depicted in Figure 7:

- *Sink-oriented solutions* centralize the schedule computation and distribution at the sink. Arisha [9] periodically computes a schedule based on traffic and battery-level information from the sensor nodes. Two slots assignment algorithms are detailed: *breadth-first search* and *depth-first search*. Breadth-first search provides contiguous time slots to the sensor nodes that share the same parent. This avoids parent nodes to repeatedly switch their radio between reception and transmission modes (which can be energy-consuming) but increases end-to-end delays. Depth-first search assigns adjacent time slots to the nodes located on the path toward the sink station. This reduces delays at the cost of more frequent radio switching at the forwarding nodes. Arisha does not propose any spatial re-use of the time slots (each link uses a unique slot network-wide), which makes this protocol hardly scalable to large WSNs.

  To alleviate this issue, the Time Synchronized Mesh Protocol (TSMP) [76] proposes to combine TDMA with FDMA. Simulta-

neous transmissions are thus possible at different frequencies within the same time slot. Frequency hopping is also considered within a same link according to the time of transmission. This makes TSMP more robust to interferences (e.g. in noisy environments), which mitigates retransmissions. The sink is in charge of computing the time/frequency slots for each given link.

- *Clustered solutions* organize the sensor nodes into clusters, the cluster head being responsible for the schedule computation and distribution. The cluster head is usually located at one-hop from the other sensor nodes in the same cluster, which avoids the collection of information over multiple hops such as in sink-oriented solutions. Nodes that would like to request a transmission slot send a notification to their cluster head during a first part of the communication window. The schedule is then computed by the cluster head and disseminated to the nodes. Slot assignments are more flexible with this approach: when a node does not have any data to transmit, its slot can be assigned to another node or used to switch off the radio at both ends of the link. In order to better spread the energy dissipation among the nodes, some protocols such as Bit-Map Assisted (BMA) [63] and Power-Aware Cluster TDMA (PACT) [74] propose to periodically alternate the cluster head among the sensor nodes.

  With the Practical Multi-channel MAC [60], cluster heads can take the initiative to change the channel employed in the cluster when too much contention or interferences are experienced between the sensor nodes.

- *Distributed solutions* avoid the overhead of centralizing the topology information as well as the schedule dissemination at the sink or at the cluster head. With Traffic-adaptive MAC (TRAMA) [80], sensor nodes learn about their two-hop neighborhood during a random access period. Each transmitter can compute the slot it owns using a hash function (with its identifier and current time as parameters). This algorithm ensures a unique transmitter within each two-hop neighborhood, which eliminates the hidden terminal problem. Once a node has computed its schedule, it is announced during the random access period in a schedule packet. It contains a bitmap of the receivers, which can then decide when to sleep or wake-up. TRAMA is however affected by a high duty cycle (12.5% without considering the transmissions and receptions). The FLow-Aware Medium Access (FLAMA) [79] protocol tries to solve this problem by avoiding the periodic information exchange between two-hop neighbors, which is transmitted upon request only.

FDMA has also been considered in distributed schemes. The Self-Organizing MAC for Sensor Networks (SMACS) [90] proposes a simple link discovery and schedule establishment algorithm. During the periodic neighbor discovery phase, a random frequency band is assigned to each new link established with a neighbor. Each link thus operates on a different frequency to avoid collisions with adjacent links. The endpoints of a link establish the transmission and reception slots within the communication window. SMACS however assumes that the underlying radio has enough available frequency bands in a way that a random selection has little chance to create links with overlapping frequencies. This weakness is tackled by the Multi-Frequency MAC for WSNs (MMSN) [118] which uniformly assigns frequencies among sensor nodes located in the same one-hop neighborhood.

Although the above-mentioned protocols try to make the schedule establishment phase energy-efficient, the short size of the time slots require network-wide and precise time synchronization [92] among the sensor nodes. As an attempt to reduce such overhead, a new category of synchronized protocols has emerged, as described below.

Common active/sleep period schemes

Sensor nodes employing a common active/sleep period wake up in a synchronized manner to send or receive data, as depicted in Figure 8.a. Sensor nodes continuously alternate sleep and active periods. During the sleep period, nodes switch their radio off, and hence save energy. Synchronization as well as frame transmissions and receptions are performed during the active period by using a contention-based scheme.

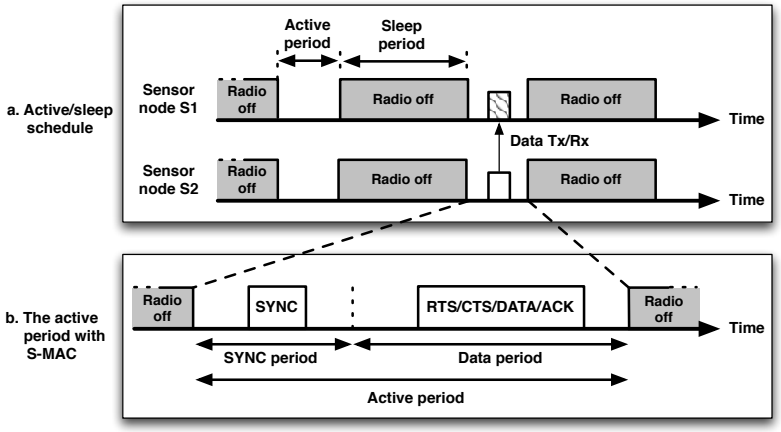


Figure 8: An example of protocol with a common active/sleep schedule: the S-MAC protocol.

In this category of protocols, S-MAC [110] is a seminal work. As detailed in Figure 8.b, the active period is divided into two consecutive phases: the Synchronization (SYNC) period followed by the data period. Although the long active period is larger than clock drifts, sensor nodes still need to periodically update their neighbors with their sleep schedule in order to prevent long-term drifts. This synchronization is performed during the SYNC period. When a sensor node boots up for the first time, it listens to the medium for a duration of at least one sleep period plus one active period. If it receives a SYNC message (which includes the address of the sender and the relative time until its next sleep period), it adopts the same schedule and disseminates it in the network by also broadcasting SYNC messages. If the sensor node does not receive any SYNC message, it broadcasts its own schedule in the network. Sensor nodes that are synchronized on the same schedule form a virtual cluster. Some nodes may follow multiple schedules at the same time if they have received different SYNC messages. These border nodes between clusters should keep their radio on during the active periods of all the announced schedules in order to prevent partitions in the network. Immediately after the synchronization period, the data period is used to send data frames between peer nodes. To handle the possible contentions in this phase, RTS/CTS messages are used for unicast packets. Acknowledgments are also employed to detect possible frame collisions.

The duty cycle of this scheme depends on the length of the sleep period compared to the active one. With S-MAC, the latter is fixed prior to the network operations, which makes it prone to idle listening and overhearing whenever the network traffic fluctuates. T-MAC [102] mitigates idle listening by prematurely and intuitively ending the active period of the sensor node if no data has been received after a timeout (Figure 9). By this way, the sleep period is increased, hence participating in more energy savings. In the same spirit, adaptive listening [110] reduces overhearing on nodes operating S-MAC by switching off their

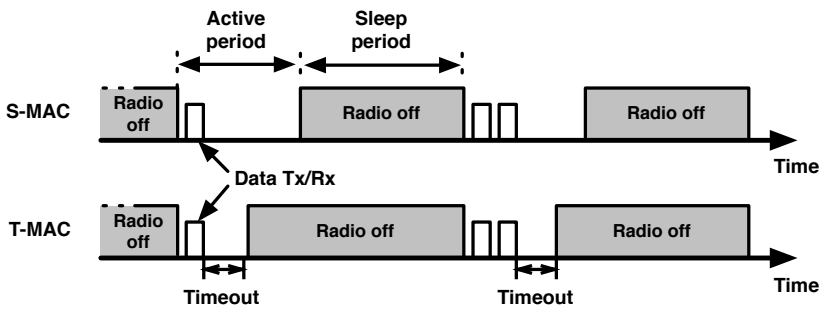


Figure 9: T-MAC reduces idle listening compared to S-MAC by allowing the node to switch off its radio earlier.

radio during transmissions in which they are not involved. For that purpose, transmission schedules and intended receivers can be learned from the RTS messages.

The IEEE 802.15.4 standard [45] also uses active/sleep periods in its beacon-enabled mode. Sensor nodes are organized in clusters, and synchronization is performed by the cluster head (the *coordinator*) using beacon messages at the beginning of the active period. A slotted-CSMA contention-access period then follows, in which nodes compete for the medium within each time slot. An optional contention-free period can come behind in order to guarantee time slots to specific nodes. An inactive period allows the sensor nodes to save energy by switching their radio off before the next beacon. Note that in non-beacon mode, a scheme very similar to CSMA/CA is employed but does not introduce energy-saving mechanisms.

Although common active/sleep period schemes do not need a network-wide synchronization protocol, the synchronization period and messages still represent a pure protocol overhead. The next Section introduces a scheme that eliminates the need for time synchronization.

1.3.2 *Preamble sampling protocols*

Preamble sampling protocols are based on CSMA. They let the sensor nodes decide of their schedule independently of their neighbors. As detailed in Figure 10, sensor nodes that employ sampling protocols always send a preamble (achieved by a carrier wave or a frame) followed by a SYNC message (which indicates the end of the preamble) before the plain data. Every node in the network periodically wakes up its radio and samples the medium. If no traffic is detected, the node switches its radio back to sleep. If a preamble is perceived, the sensor remains awake to receive the trailing data, which can optionally be acknowledged. The receiver then goes back to sleep. The preamble thus ensures that the neighboring nodes will be ready to receive the data. For that purpose, the Preamble Length (PL) must indeed be longer than the Sampling Period (SP) on the nodes.

Preamble sampling techniques have been initially combined with traditional wireless network protocols, such as ALOHA in [32] and CSMA in [44] in order to enable low-power operations of these protocols. In the WSN world, Berkeley MAC (B-MAC) [77] is a seminal work on the subject. It operates as depicted in Figure 10 but defines a new algorithm that improves the Clear Channel Assessment (CCA) operations. The goal of the CCA is to decide whether a channel is busy or not. It is usually based on thresholding, which compares the power of the measured signal to a noise floor. This can however lead to a significant

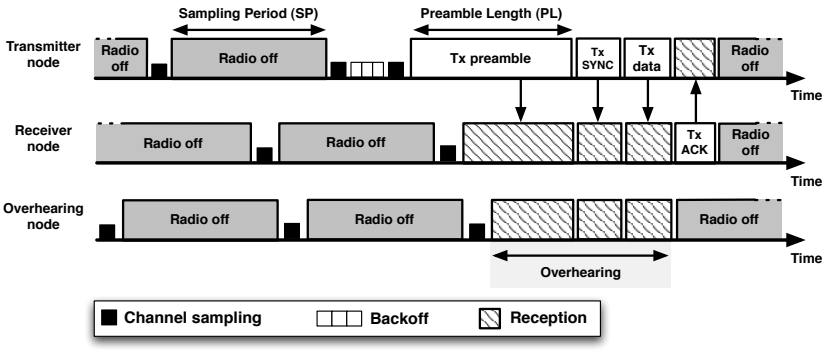


Figure 10: Preamble sampling protocols perform a regular channel sampling in order to detect a preamble, which always precedes the data.

number of false positives due to the instability of such measures over time in wireless environments [43]. With B-MAC, the channel is sampled multiple times. Whenever a sample is significantly below the noise floor (such sample is called an *outlier* value), the channel is considered as clear. If no outlier is found after five consecutive samples, the channel is considered busy. This mechanism guarantees better results because outliers are infrequent when a valid signal is transiting over the channel. Finally, each node also updates the noise floor over time in order to adapt to possible changes in the ambient noise. A performant CCA algorithm improves energy-efficiency by mitigating collisions. Note that B-MAC refers to the preamble sampling technique as Low Power Listening (LPL). This acronym will be regularly used in the remainder of this document.

LPL achieves large energy savings by removing the need for a time synchronization protocol among the sensor nodes. The preamble is employed as a loose synchronization method of the sleep-listen period. Receivers can switch off their radio during long periods, waking up only during a short time to sample the medium. Idle listening is thus reduced on the receiver side, but the preamble increases the transmission costs on the sending node. Low duty cycles can be achieved with large Sampling Periods (hence longer sleeping phases). This however mandates senders to use longer preambles, which augment the energy dissipation upon transmissions. This also results in more overhearing on recipients, particularly on the nodes that are not the destination of the data, as depicted in Figure 10. The energy dissipation caused by the preamble makes this scheme more suitable for nodes that seldom collect and report data. Multiple solutions have thus been suggested to reduce the Preamble Length to the minimum while keeping long sleeping periods.

The Sparse Topology and Energy Management (STEM) [85] protocol uses two wireless transceivers: one is employed as a

wake-up channel, and the other one for data communications. The data radio is switched off as long as no synchronization has been performed through the wake-up radio. The STEM-T variant of the protocol uses LPL on the wake-up channel as a way to synchronize the data channel with its peers, while minimizing the energy consumption of the transceiver. STEM-B optimizes the wake-up channel by using a succession of beacon messages (instead of a simple tone) which include the address of the intended receiver. Upon reception, the receiver can shorten the succession of beacon messages by sending an acknowledgment that indicates it is ready for the communication. The use of two transceivers however raises multiple issues in terms of hardware costs and energy consumption.

The concepts proposed by STEM-B have thus been adapted to operate on a single channel. The X-MAC [17] protocol divides the long preamble of B-MAC into smaller frames, each containing information about the destination of the ensuing data (Figure 11). A small interval is introduced between each preamble frame in order to let the receiver acknowledge its reception. Other nodes can switch off their radio as soon as they detect that they are not the intended receiver. The data frame is transmitted by the sending node as soon as the acknowledgment is received. As depicted in Figure 11, X-MAC mitigates both overhearing on non-recipient nodes and long preamble transmissions on senders. Similar techniques have also been used in the Transmitted Initiated Cycled Receiver (TICER) [64] protocol and Multimod-Hybrid MAC (MH-MAC1) in asynchronous mode [15]. This idea of a strobed preamble is however suitable only in unicast communications. Whenever broadcast packets must be sent, the whole preamble still needs to be transmitted in order to ensure that all neighbors are correctly synchronized on the same schedule as the sender.

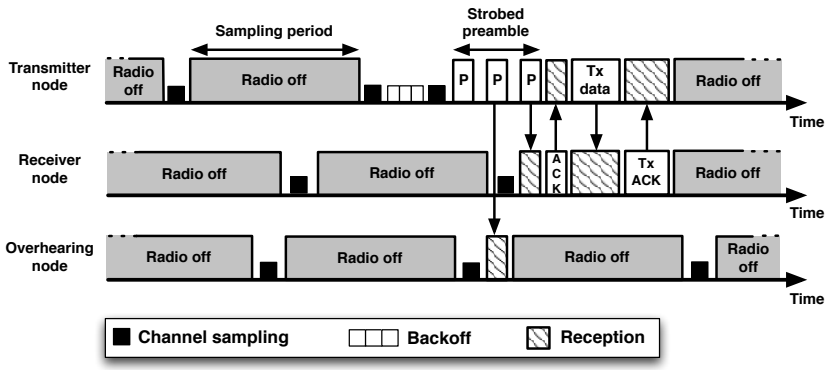


Figure 11: The use of a strobed preamble reduces the costs on the transmitter as well as overhearing on recipients.

In broadcast communications, overhearing can be reduced when nodes detect a preamble by allowing them to put their

radio to sleep until the data will be effectively transmitted on the medium. Micro Frame Preamble MAC (MFP-MAC) [11] and Wake-Up Frame (WUF) [88] introduce in the strobed preamble the number of remaining frames before the data. Neighbor nodes that overhear one of them can thus switch off their radio during the remainder of the preamble.

By knowing the wake-up time of its peer, a sensor node could further reduce the size of the preamble. With the Wireless Sensor MAC (WiseMAC) [33] protocol, each sensor node learns the time of the next channel sampling of their neighbors during every data exchange as part of the acknowledgment message. The transmitter then only needs to send a small preamble (to be resilient to clock drifts) around the wake-up time of its correspondent. If the sender node does not know such schedule yet, it employs a full-length preamble. Similarly, the Synchronized Wake-Up-Frame (SyncWUF) [8] protocol combines WiseMAC and WUF in order to further reduce the energy dissipation caused by the transmission of a preamble. CSMA with Minimum Preamble Sampling (CSMA-MPS) [66] combines the principles of WiseMAC and X-MAC, and proposes to replace the strobed preamble with a repetition of the data frame. This improves reliability because the receiver acknowledges the preamble frame only if the data it contains has correctly been received.

1.3.3 *Hybrid protocols*

Hybrid protocols merge the concepts of the preamble sampling and synchronized schemes in order to get to best of each. Such combinations can be particularly appealing to address scenarios with variable traffic conditions.

The Zebra MAC (Z-MAC) [82] protocol uses B-MAC network-wide as long as the contention level remains low. When the load increases, sensor nodes switch to a slotted scheme as an attempt to reduce collisions during high contention periods. Slot distribution is performed using a distributed algorithm (DRAND [83]). Sensor nodes are allowed to transmit during any of the time slots, but the slot owner has a smaller contention window which gives it earlier chance to transmit. Carrier sense is performed before accessing the medium to avoid collisions inside a slot. Z-MAC can experience schedule drifts, as reported in [2], which mandates all the nodes to periodically execute DRAND hence reducing its energy-efficiency.

In convergecast models, the sensor nodes report their data toward a sink. In large sensor networks, the nodes located around the sink would likely experience congestions as they have to forward packets coming from the whole network: this is the *funneling effect*. In order to address this bottleneck issue, the Funneling-

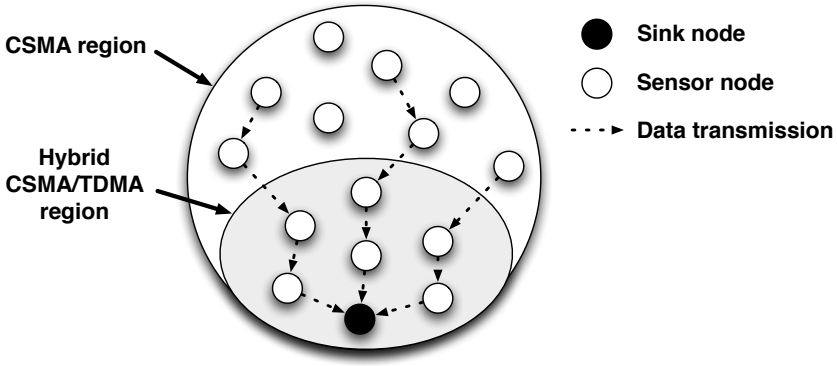


Figure 12: Funneling-MAC combines different schemes in various locations of the network.

MAC [2] protocol suggests the use of an hybrid CSMA/TDMA algorithm around the sink while a CSMA scheme (e.g. B-MAC) is implemented in other places (Figure 12). The sink regularly sends a beacon, at a varying power, to create the region in which the hybrid scheme will be operated. The power used to send the beacon is calculated according to the measured traffic and losses at the sink. Nodes that receive such a message separate their communication window into a CSMA part and a TDMA one. The TDMA slot allocation is centralized at the sink and broadcasted in a schedule packet. The sink is aware of each path in the hybrid region (thanks to informations obtained from the data packet headers), and thus can distribute slots according to the load on each path.

Hybrid techniques can also be employed to further reduce the energy consumption of a particular scheme. For example, the Scheduled Channel Polling (SCP) [111] protocol uses a common active/sleep scheme such as in S-MAC, and proposes to mitigate idle listening during the active period by using preamble sampling techniques. The overhead of the synchronization period is reduced by piggybacking the schedule in the data frames. In order to mitigate overhearing on the receiver side, Crankshaft [39] defines a receiver-oriented slotted scheme: it schedules the receivers instead of the senders. The slot a node listens to is determined by its MAC address. During that slot, several sensor nodes may send a message to the peer that owns the slot. In that case, the contention is resolved by using a short preamble before the transmission, which secures the slot. An additional set of broadcast slots, in which all receivers listen, allows the transmission of broadcast frames.

Category of protocol	*Main characteristic*	*Protocol name*
Slotted	Sink-oriented	Arisha [9], TSMP [76]
	Clustered	BMA [63], PACT [74], Practical multi-channel MAC [60]
	Distributed	TRAMA [80], FLAMA [79], SMACS [90], MMSN [118]
Common active/ sleep period	Seminal	S-MAC [110]
	Early sleep	T-MAC [102], Adaptive listening [110]
	Standard	IEEE 802.15.4 [45]
Preamble sampling	Seminal	B-MAC [77]
	Multiple radios	STEM [85]
	Strobed preamble	X-MAC [17], TICER [64], MH-MAC1 [15], WUF [88], MFP-MAC [11]
	Synchronized	WiseMAC [33], SyncWUF [8], CSMA-MPS [66]
Hybrid	Slotted + sampling	Z-MAC [82], Crankshaft [39], Funneling-MAC [2]
	common active/ sleep + sampling	SCP [111]

Table 4: Summary of main energy-efficient MAC protocols for WSNs.

1.4 CONCLUSION

The constraints induced by the WSNs have motivated researchers to propose new schemes to access the medium. Especially, achieving low power communications has been the major challenge in the last decade, in contrast with delay improvements or fairness in traditional wireless networks. As an attempt to mitigate the energy dissipation mostly due to idle listening, new categories of protocols have emerged and are summarized in Table 4.

Synchronized schemes schedule periodic communications and sleeping periods among the sensor nodes. Preamble sampling protocols operate in an asynchronous manner which allows for more flexibility while being resilient to clock drifts. Hybrid protocols combine the characteristics of the previously mentioned schemes in order to propose better performances in networks where the traffic conditions differ in space or time.

In this Chapter, we have particularly focused on energy-efficient contributions. In Chapter 2, we will discuss the suitability of each scheme with regard to dynamic WSNs, and detail the solutions that have been proposed so far to better handle such networks.

2 RESEARCH TRENDS AND FORESEEN ISSUES

Contents

Along with the continuous advances in embedded systems, new WSN use cases have emerged. Thanks to the miniaturization and cost reductions, we can foresee large-scale and dense deployments that would aim at operating unattended for long periods, while the network evolves in space and time.

In this Chapter, we will particularly detail two aspects of such evolutions: node mobility and changes in the traffic pattern. Although these characteristics have already been studied in the past, we question the viability of the proposed solutions when applied to large-scale, dense networks. Indeed, the fact that large quantity of sensor nodes co-exist within the same one-hop radio neighborhood would bring new shortcomings when accessing the medium. After detailing these issues, we will overview the MAC layers that were designed to sustain such dynamic WSNs. These solutions adapt the schemes detailed in Chapter 1 by introducing new mechanisms while keeping in mind energy-efficiency. We will especially discuss which of the synchronized, preamble sampling or hybrid schemes would best address the presented scenarios, and motivate the reasons for our contributions.

2.1 NEW CHALLENGES IN FUTURE DEPLOYMENTS

Among the large variety of deployments operated so far [84], a few stand out and try to address more challenging issues. For example, the use of mobile sensors has been reported in [6,98,115]. More recently, the deployment exposed in [19] made both the event-driven and time-driven data collection schemes co-exist in the same network, each having their own requirements in terms of data rates and packet delivery. However, as pointed

in [51], these deployments are of relatively small scale (tens of sensors divided into small patches) and achieve hardly up to 6 communication hops. Furthermore, most of them are operated from a few days to a few weeks, which usually relegates energy matters in the background. So far, these constraints remained accessible enough to be satisfied by simple MAC protocols and energy management models.

In the near future, contemplated scenarios will involve more sensor nodes in multi-hop topologies, while expecting to last months or even years. In addition, both the topology of the network and the way it delivers the information would evolve during the lifetime of the deployment. We will particularly study two aspects of such dynamic networks at the MAC layer, as detailed below.

Accessing the medium in a mobile environment

Mobility is foreseen as a very likely solution to expand the network coverage [114], improve the routing performances or the overall connectivity [26]. Sensor nodes may also move from their locations because of the environment in which they operate (e.g. ocean or air). The necessity to collect fine-grained information will certainly make the mobile sensors traveling across dense topologies of fixed sensor nodes. In that context, the mobile nodes need to be integrated in the communication schedule of their neighbors. This can particularly be a problem when a schedule has already been decided among the fixed nodes. Beside, the data collected and reported by the mobile nodes may be crucial in some cases, such as target tracking or surveillance applications as depicted in [62]. The mobile sensors should be able to access the medium whatever the level of the contention on the medium. Furthermore, such access should be performed at the lowest delay. By design, some MAC schemes for WSNs may face difficulties to address these scenarios. We will detail these possible shortcomings as well as the existing solutions in Section 2.2.

Multiple data collection schemes and burst transmissions

The way the information is collected and reported toward the sink can evolve with time. For example, let us consider a large sensor network operating in an event-driven fashion. Upon an event of interest, the node located around the spot starts reporting information about the incident in a time-driven manner toward the sink, at a certain rate. Such events could happen in a-priori unpredictable locations. They could be triggered for example by the detection of a subject in a surveillance application (such as illustrated in [62]), an observed value reaching the threshold in a

monitored scene (for example seismic events [106] or structural monitoring [49]). In the context of mobility, this could translate in data muling, i.e. the dissemination of information collected by a mobile sensor from stationary nodes located out of range of the network. The traffic pattern changing from sporadic to burst transmissions can degrade the energy-efficiency of the protocol being operated in the network. Depending on the routing policy (e.g. sink-rooted tree in most of existing scenarios [14, 19]), only a subset of the deployed nodes will participate in relaying the reported data toward a sink station. As a result, only a part of the network would need to adapt its behavior according to the constraints of the moment. Some versatile schemes already offer the possibility to adjust their MAC parameters in order to operate efficiently under multiple scenarios. We will explain in Section 2.3 to what extent versatility can be employed, and how these protocols could be made truly auto-adaptive.

Along with the introduction of the available solutions in the next Sections, we will clarify whether they address the shortcomings raised by the above-mentioned scenarios.

2.2 HANDLING MOBILITY AT THE MAC LAYER

We overview the potential design problems that could refrain the MAC schemes detailed in Chapter 1 to operate in a mobile environment, as well as the main contributions that have been proposed to mitigate these issues.

2.2.1 *Synchronized protocols*

As detailed in Chapter 1, synchronized protocols comprise slotted and common active/sleep period schemes.

Slotted schemes

Slotted schemes present several shortcomings resulting from their reliance on the network topology and time synchronization. The knowledge of the topology is required to establish a collision-free schedule. This characteristic makes these protocols hardly operational in mobile environments. Indeed, every time a new sensor node wishes to leave or participate in the communication, the neighboring nodes can barely change their schedule without computing and distributing the time slots again (Figure 13).

As an attempt to alleviate this issue, Flexible MAC (FlexiMAC) [61], Eyes MAC (EMACs) [103] and EAR [90] allow fixed sensors to offer momentarily to mobile ones the time slots they own. This negotiation is performed using control messages, which consumes additional energy. Furthermore, the number of time slots in the

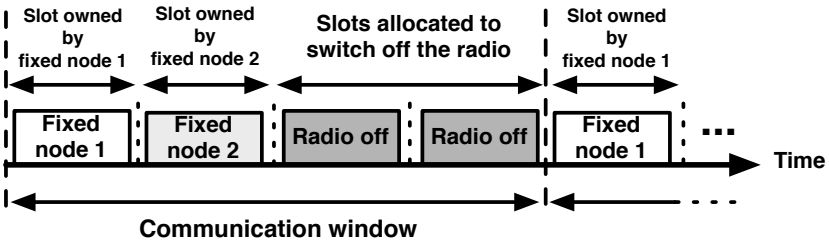


Figure 13: Mobile nodes cannot integrate the communication unless the time slots are distributed again.

communication window being bound, such proposal remains hardly scalable to a large number of mobile nodes. Mobility-adaptive MAC (MMAC) [5] thus proposes to adapt the size of the communication window according to the level of mobility in the network. This size is inversely proportional to the level of mobility and is periodically computed by the cluster head before being broadcasted in the network. Yet, MMAC assumes that the sensor nodes are aware of their location and able to estimate their trajectory. This is difficult to achieve without the overhead of a localization protocol or additional hardware such as a Global Positioning System (GPS).

These solutions can make slotted protocols more flexible to little changes in the network topology. Still, they require a tight synchronization in order to maintain a common schedule among the sensor nodes. This constraint remains hard to alleviate in large-scale multi-hop networks, and the use of additional synchronization protocols [92] or hardware induces large overheads in terms of control messages or costs. As a consequence, we will not consider slotted protocols as a realistic solution to handle mobility at large scale in WSNs.

Common active/sleep period schemes

Protocols based on a common active/sleep period scheme can hardly integrate mobile sensors in their communication scheduling algorithms. For example, the periodic synchronization which is essential in S-MAC may prevent a mobile node to send or receive data if it does not know the schedule employed in the virtual cluster where it is located. In order to learn it, the mobile sensor must receive a SYNC message, hence listen to the medium at least during one active period plus a sleep period, which increases idle listening on the mobile node (Figure 14). Furthermore, the time spent to re-establish a schedule postpones the transmission of the queued frames, which impacts the communication latency between a mobile and its peers. To solve this, Mobility-aware S-MAC (MS-MAC) [75] extends S-MAC and suggests to increase the frequency of the synchronization period in the cluster according

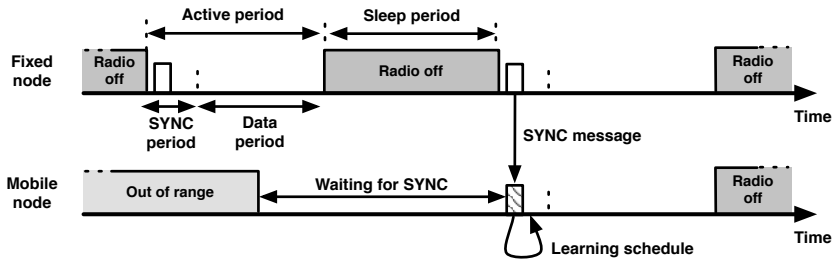


Figure 14: With S-MAC, mobile nodes need to learn the schedule used in the cluster before being able to take part in the communications.

to the speed of the mobile sensor. In practice, the received signal level used as a mobility indication does not provide a fair accuracy to evaluate proximity, as detailed in an empirical study in [43].

The IEEE 802.15.4 standard requires each node to be associated with their coordinator. The association is maintained over time thanks to beacon messages. In the non-beacon-enabled mode, beacons are sent by coordinators only upon request. A moving node would thus assume that its association is always preserved even though it has moved away from its coordinator, unless it continuously sends beacon requests. As already mentioned in Chapter 1, this mode does not allow for much energy savings. Instead, with the beacon-enabled mode, a node considers itself as an orphan when it does not receive a predetermined number of beacons within a specific time interval. It can trigger the *orphan device realignment procedure* [45] whose purpose is the re-association of the node with a new coordinator. This approach however imposes a large inaccessibility time due to the attempt to contact the previous coordinator. According to the node speed, it may even move to another cluster before the end of the association with its new coordinator, as reported in [113].

The common sleep/active period schemes require the establishment and maintenance of the schedule using synchronization messages. In order to avoid long disconnections experienced by mobile sensors, more periodic synchronizations must be performed, which ineluctably increases the control overhead and its resulting energy cost.

2.2.2 *Preamble sampling protocols*

Protocols relying on preamble sampling techniques do not rely on a cluster nor a network-wide schedule, which makes them less complex to operate in dynamic networks: no schedule dissemination nor time synchronization needs to be performed. In theory,

every node may be able to emit at anytime on the medium as each can choose its active schedule independently from the other.

The use of a preamble however reduces the channel availability, and therefore increases the competition among the nodes. According to the network density and frequency of the data collection, collisions can be frequent and would imply retransmissions, which worsen the contention while consuming even more energy. When traveling across such congested areas, mobile sensors would experience long medium access delays. They also increase the competition of the shared medium, as they constitute one more sender in the area where they travel. Considering that a moving sensor would not be in the vicinity of the perceived event forever and may not have the capacity to store much data in queue, the time needed to access the channel must be decreased. Furthermore, the medium access delay combined with the node mobility may prevent the mobile sensor from reaching its next hop. Indeed, the expected peer (chosen at the routing layer) may already be unreachable when the data packet is actually sent on the medium, as depicted in Figure 15.

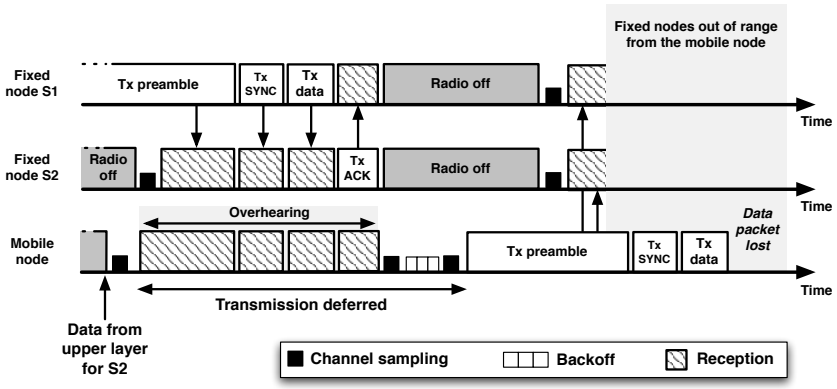


Figure 15: In contented networks, a mobile node may not have the time to reach its expected next hop.

Although the use of a strobed preamble with X-MAC or the synchronized preamble of WiseMAC can mitigate the medium access delay, several other shortcomings deserve further investigations. First, when a sensor node is not continuously in range of the moving node that emits the preamble, it may not detect any signal when sampling the channel. As a result, its radio may be switched off when the correspondent sends the data. It is all the more true with mobile-to-mobile communications. Also, these solutions always have to fallback to the full-length preamble in broadcast communications.

Preamble sampling protocols do not require any topology knowledge nor time synchronization, and can operate in a completely distributed manner. This makes them very robust with

regard to topological changes as well as scalable to large networks. Still, the above-mentioned shortcomings are worth considering. Hybrid protocols have been suggested to tackle some of these issues (see Section 2.2.3), but these solutions inherit the drawbacks of the slotted protocols. Instead, it would be interesting to tackle them without adding control messages in order to keep the benefits of sampling protocols. As an attempt to better understand these issues in large-scale and dense networks, we will further study the behavior of the preamble sampling protocols on mobile sensors in Chapter 3.

2.2.3 *Hybrid protocols*

As exposed in the previous Sections, mobile sensors may face difficulties to access the medium, either because they do not own a communication slot, or due to the high contention level in the portion of the network they travel across.

In order to reserve a fraction of the channel for the mobile sensors, the Mobility Adaptive Hybrid MAC (MH-MAC2) [78] protocol proposes to divide the communication window in two parts: one for the fixed nodes, with a medium access controlled by a slotted protocol, and a second one using a contention-based protocol for the mobile sensors. The length of each window is dynamically adapted according to the ratio of mobile nodes to fixed ones. Still, communication between both types of sensors does not seem possible (as each uses a different part of the window), unless through an intermediary one that continuously participates in both phases. The Mobility Aware and Energy Efficient MAC (MEMAC) [108] protocol proposes a similar separation of the communication window in order to differentiate between data and control messages. Long data frames are assigned a time slot in the scheduled access period while short control frames are sent in the random access period. These control messages allow the sensor nodes to join or leave clusters as well as request communication slots.

As already discussed, these hybrid protocols inherit the downsides of the synchronized schemes. The synchronization of the network as well as schedule computation and dissemination through control messages represent a significant overhead in terms of energy costs.

2.3 VERSATILITY AND AUTO-ADAPTATION

Evolution of the topology (due to mobility, death of nodes or re-deployment strategies) can be one of the causes of substantial changes in the data collection scheme or in the traffic load. While guaranteeing the medium access at the first hop is indeed im-

portant, it is also crucial to ensure that the global performances do not suffer from such network dynamics. To some extent, MAC protocols for WSNs propose versatility features to operate in an energy-efficient manner under various conditions. The below Section will discuss whether this is enough to maintain decent performances in the network.

2.3.1 *Limitations of versatility*

Each of the synchronized and preamble sampling protocols offer some flexibility in terms of configuration prior to their deployment and operation in the network. For example, the communication window of slotted protocols is usually calculated as a tradeoff between the desired throughput, latency and duty cycle. However, when one of these changes, the schedule needs to be computed again to remain performant. What was true for topological modifications also holds when the traffic varies: when nodes have more data to send, time slots are missing. Conversely, if sensor nodes stop sending traffic, time slots are wasted.

Protocols based on a common active/sleep period can also be configured prior to their deployment. The length of the active period depends on the expected traffic, and the sleep period determines the duty cycle of the node. A short active period reduces idle listening at the cost of an increased contention. On the contrary, a long one reduces contention but increases idle listening. Once fixed, any changes in the traffic pattern can degrade the performances of the protocol: whenever the load decreases, the sensor nodes would spend more time in idle listening in the active period. On the opposite, high traffic would cause more collisions if the active period is not large enough. For example in [18], authors have deployed a soil moisture monitoring system during rainstorms. While no rain is detected, all the sensor nodes in the network operate in a time-driven fashion. Moisture information is reported once a day to the sink. Upon an event triggered by a rainstorm, measurements are collected at a higher frequency (every minute) on each sensor node. This change in the collection interval lasts up to two hours after the end of the rainfall. After that, the sensor nodes go back to the original data collection period. All the sensor nodes are pre-configured with the S-MAC protocol and a 12-second sleep period. With this value, the network can sustain the load generated during rainfalls, when the data is reported at a high frequency. However, it causes a waste of energy (due to idle listening) in dry periods as data is sent at a much lower frequency. In the latter case, the cycle length could be increased in order to extend the radio sleep time. Still, local decisions to operate a different cycle could lead a node to fail synchronizing with its neighbors, as detailed in Figure 16.

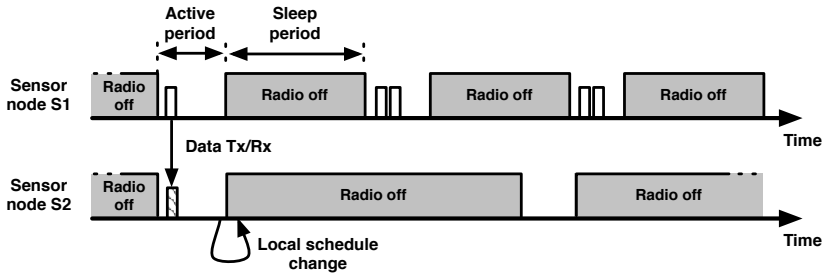


Figure 16: Two nodes operating S-MAC with a different active/sleep period would fail communicating with each others.

Preamble sampling protocols can also be configured prior to a deployment. Especially, the LPL can be tuned through the Sampling Period (SP, which determines the duty cycle) and the Preamble Length (PL, which must be as long as the SP). Usually, long preambles are employed when little traffic travels in the network. On the contrary, when the traffic increases, shorter preambles should be used in order to decrease the transmission cost on the sender, and to increase the channel availability. It is thus necessary to adapt the LPL parameters according to the current traffic load. The B-MAC protocol has enforced and extended this idea by defining a set of interfaces made available to the upper layers. In addition to the SP and PL, it exposes several other configurable parameters, such as the backoff length and the use of link-layer acknowledgments, which can be decided on a per-frame basis while the network is operating. One can then configure B-MAC with a set of parameters suitable for the deployment characteristics. For example, reliability can be achieved by activating acknowledgments. Similarly, the desired throughput and per-hop latency can be adjusted by modifying the LPL parameters. It however requires that the application and associated data traffic are well foreseen. Furthermore, B-MAC does not specify how these parameters can be configured on-the-fly, while the network is running. It is obvious that a misconfiguration on one node could isolate it from its neighbors. A simple example is given in Figure 17, where different LPL values (i.e. SP and PL) are used on distinct nodes. Instead, it is preferable to fix the same set of values on all the nodes prior to the deployment.

On the software side, it is interesting to note that versatility can be achieved by wiring different modules in order to build a suitable communication stack, as detailed in [70]. This mechanism could be used to dynamically switch the MAC layer according to the needs, similarly to what MiX-MAC [69] achieves. Although rewiring components on-the-fly is possible in today's WSN operating systems (e.g. with Contiki [29]), it requires substitute components stored in memory. Furthermore, it restricts the level of adaptation that can be achieved by the protocols. Instead, a uni-

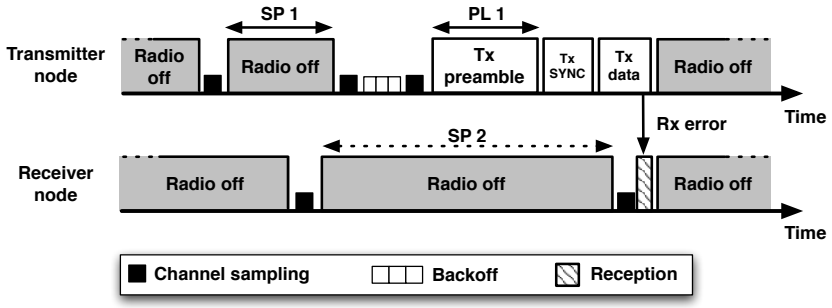


Figure 17: Different LPL configurations can lead to synchronization issues, for example if the PL on the transmitter is shorter than the SP on the receiver.

fied MAC would be more efficient while enabling more elaborated adjustments.

To conclude, versatile protocols enable the pre-configuration of a same protocol for various scenarios. They however remain performant provided that the traffic does not change over time. In order to address efficiently scenarios with time-varying or spatially non-uniform traffic loads, MAC protocols would need to adapt their behavior dynamically. It therefore excludes any prior installation of a specific MAC layer on the sensor nodes. Instead, the way to access the medium should be adjusted according to the constraints of the moment. Ideally, protocols would have to enforce local decisions to adapt to the network conditions, while ensuring fair performances in the whole network. At the MAC level, this translates into energy-efficient auto-adaptive protocols that could perform well under various loads while avoiding deafness among neighbors.

2.3.2 *Auto-adaptive algorithms*

Because an unfortunate modification could isolate a node, or even create a partition in the network, auto-adaptive methods should be designed to ensure a fully automated setup and composition of the medium access protocol while ensuring layer-2 communication among the sensor nodes. We will detail hereinafter the major proposals that dynamically adapt the duty cycle of a node according to the traffic variations in the network.

Slotted schemes

The solutions designed for mobile scenarios [5, 61, 90, 103] and detailed in Section 2.2.1 could be employed to release or reserve time slots according to the traffic load. In addition, the Pattern MAC (PMAC) [117] protocol defines an algorithm to compute the schedule of a node according to its own traffic and the one of its

neighbors. Especially, the number of sleep slots is dynamically increased when the traffic decreases, and conversely. Connectivity among the nodes is guaranteed by a set of rules that enforces the schedule of each node according to the one of its peer. However, as previously discussed, slotted protocols suffer from scalability issues and large control overheads to compute and distribute time slots among the nodes.

Common active/sleep period schemes

In order to adapt to traffic variations, the size of the active period should be flexible and dynamically adapted to absorb the traffic without provoking collisions nor spending too much time in idle listening. For that purpose, Dynamic S-MAC (DSMAC) [65] improves S-MAC with a dynamic duty cycle mechanism. Each node initially starts with the same schedule, but when it notices a higher traffic load it can independently increase its duty cycle by adding an extra active period in the middle of its current sleep period (Figure 18). The new schedule is announced in a SYNC message, whose receivers can decide whether they adopt it or not. The nodes that do not follow the announced schedule can still communicate with the other ones during the initial active period. When the traffic decreases, nodes can come back to their initial duty cycle by removing the added active periods.

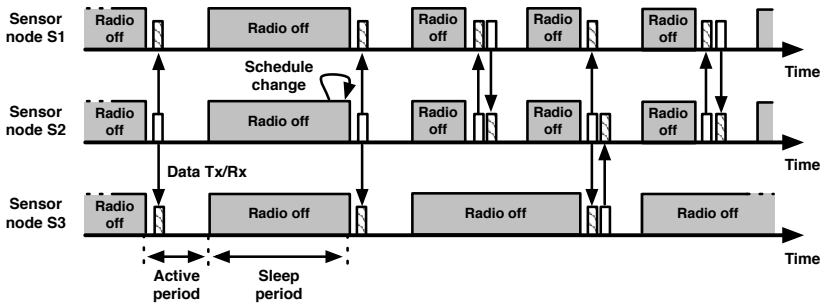


Figure 18: DSMAC operations: S2 changes its schedule and advertises it in a SYNC message. S1 decides to adopt the same schedule, but S3 remains on the old one. Still, S3 is able to communicate with S2 in the initial active period.

Likewise, the Utilization-based MAC (U-MAC) [109] protocol allows each sensor node to set its own duty cycle. The basic idea is similar to T-MAC (switching off the radio earlier than initially scheduled) but U-MAC replaces the timeout with an utilization function, which computes on each node the ratio of transmissions and receptions to the size of the active period. A low value means high idle listening: the node reduces its next active period and advertises its new schedule in a SYNC message. Neighbor

nodes do not adopt that schedule but the fact that they know the expected sleep time helps to avoid deafness among the nodes.

Preamble sampling schemes

The major challenge with preamble sampling protocols is to tune the LPL without isolating nodes. For that purpose, auto-adaptive algorithms that adjust the LPL values must guarantee that the sender's PL matches the receiver's SP (Figure 19).

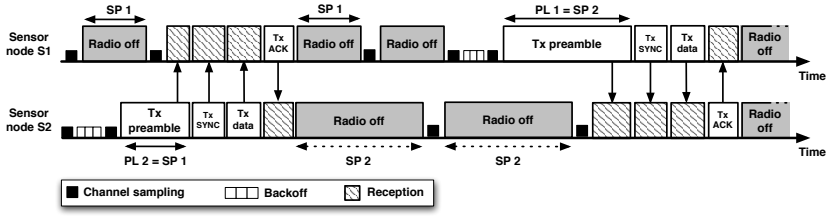


Figure 19: S1 and S2 uses different SP values. Communication between both nodes is guaranteed as long as the sender's PL matches the receiver's SP.

The Energy-Aware Adaptive LPL (EA-ALPL) [48] algorithm employs the B-MAC interfaces to modify the LPL values. Each sensor node periodically announces its SP through routing update messages (EA-ALPL makes the assumption that a pro-active routing protocol is used). This SP is obtained with an heuristic based on the current and past forwarding loads. Neighbor nodes can then adjust the PL value to match the SP of the sender. This algorithm however relies on routing protocol messages to advertise the changes in the LPL configuration. Losing one of them (e.g. due to a collision) may definitely isolate the sender node.

In [68], the authors present an approach to dynamically adjust the duty cycle of a node so that the traffic rate can be sustained while minimizing the energy consumption. They use control theory [50] to adapt the SP and PL of the node. Each sensor node can compute these values independently from their neighbors. Peers are presumed to converge to similar configurations, hence no control message is needed between the nodes. However, packet losses can be experienced as long as the communication endpoints do not use the same values. This approach has been simulated between two nodes only, we can thus expect lower performances in large-scale, multi-hop networks.

Authors of the X-MAC proposal also present in [17] an adaptive algorithm to adjust the LPL values on the sender and the receiver. This algorithm requires the knowledge a-priori of the probability of receiving a packet in any given interval. This can introduce a certain latency in the establishment of a different schedule on the routing path, as the computation of new values would first require the reception of a few packets at a higher or lower

rate. Furthermore, the authors do not explain how configuration changes could be reflected on the neighboring nodes, which is required to avoid isolating nodes.

2.4 CONCLUSION

In this Chapter, we have detailed the various contributions that aim at improving the medium access in dynamic WSNs. We have particularly focused on two aspects: the integration of a mobile sensor node in the communication schedule of its neighborhood, and the adaptation of the schedule in order to sustain variable traffic loads. A summary of the overviewed protocols is given in Table 5.

Slotted protocols are hardly scalable to large sensor networks. The network-wide schedule computation and distribution, as well as time synchronization represent a significant signaling overhead. This category of protocols also lacks flexibility with regard to dynamic WSNs. As hybrid protocols inherit from slotted ones, the same drawbacks partly apply to these schemes.

The fact that no medium reservation is performed a-priori makes contention-based methods more adapted to deal with network dynamics. Protocols relying on a common active/sleep period however require additional control messages when dealing with dynamic scenarios. This can constitute a large part in the overall energy dissipation, especially under light traffic conditions. On the contrary, preamble sampling schemes have the key advantage that senders and receivers can be completely decoupled in their duty cycles. Neither global synchronization nor topology knowledge is required, while they operate in a completely distributed manner. They have furthermore proved their energy-efficiency compared to synchronized protocols [77].

Category of protocol	*MAC for mobile scenarios*	*Auto-adaptative MAC or algorithm*
Slotted	FlexiMAC [61], EAR [90], EMACs [103], MMAC [5]	PMAC [117]
Common active/ sleep period	MS-MAC [75]	DSMAC [65] U-MAC [109]
Preamble sampling	None	EA-ALPL [48], X-MAC [17] Control theory [68]
Hybrid	MH-MAC2 [78] MEMAC [108]	None

Table 5: Summary of main MAC protocols addressing mobility and auto-adaptation of the duty cycle.

Preamble sampling schemes thus appear as a likely solution to handle mobility. A few methods combine them with slotted techniques in order to reserve a portion of the channel to the mobile sensors (e.g. with MH-MAC2). However, in order to scale to large and dense topologies, we advocate the use of a pure preamble sampling scheme. As an attempt to better understand the issues that could appear in a mobile scenario with this category of protocols, we will especially study the behavior of B-MAC in a mobile environment in Chapter 3. This first analysis has led us to design the *Machiavel* protocol, that we will detail in Chapter 4. We have then extended our proposal by applying it to X-MAC, in order to demonstrate that the concept behind Machiavel can also be applied to optimized preamble sampling protocols. This enhanced version called *X-Machiavel* is detailed in Chapter 5.

Initially, preamble sampling protocols are more suitable in scenarios with sparse traffic (as sending a preamble results in a significant overhead on the transmitter). Combined with auto-adaptive algorithms, global performances can be improved when the data traffic changes over time. Still, current solutions such as EA-ALPL are not satisfying to our minds: they rely on control messages or complex algorithms. These schemes trade energy to adapt the schedule of the nodes, which is not acceptable given the energy constraints. In order to better understand the issues raised by a pre-configured protocol, we first evaluate in Chapter 6 the overall performances of a non-optimized (B-MAC) and an optimized (X-MAC) pre-configured preamble sampling protocol when operated in a large-scale dynamic network. We then advocate the need for a simpler and more energy-efficient auto-adaptation method: our algorithm detailed in Chapter 7 does not require any extra control messages and it can be adapted to various preamble sampling protocols.

Part II

ACCESSING THE MEDIUM IN A MOBILE WSN

3

MOBILITY ISSUES WITH SAMPLING PROTOCOLS

Contents

In this Chapter, we would like to emphasize the potential issues that may prevent a mobile sensor operating a preamble sampling protocol to correctly integrate the medium access schedule of its neighboring fixed sensors in a large-scale WSN. We have chosen to work with the B-MAC [77] protocol. It is one of the reference preamble sampling protocols in WSNs, and has already been successfully used in real deployments, such as in [93, 99]. The goal of this preliminary study is to better understand the possible shortcomings on a simple, non-optimized protocol. Further investigations on an optimized one (X-MAC) will be performed in Chapter 5.

3.1 SIMULATION ENVIRONMENT

Large-scale experimentations have been performed using simulations. We have used the WSNet software [21], which is a wireless sensor network simulator that provides various interference and modulation models, as well as the possibility to configure the radio medium accurately. Energy consumption models are also available in order to represent per-node energy depletion during the simulation. Its modular aspect makes it very easy to extend to new models and protocols. In order to highlight the issues that can arise with B-MAC in a mobile environment, we have developed an implementation of the protocol for WSNet. This implementation has been contributed and integrated to the WSNet software in July 2009.

The simulation scenario consists in a mobile sensor moving in a network whose density and data communication frequency of the fixed nodes are gradually increasing. Table 6 exposes the details of our simulation environment. 10 to 400 fixed nodes (which corresponds to an average number of neighbors from

Simulation parameter	*Value*
Topology	Regular square grid (20x20 m), mobile and fixed sensors distributed randomly for each simulation
Mobile sensors	1
Number of fixed sensors	10, 20, 30, 50, 75, 100, 150 200, 250, 300, 350, 400
Data sending period	Mobile sensor: 1 s
	Fixed sensors: no data sent, 4 s, 2 s, 1 s
Data packet size	18 Bytes
Mobility model	Billiard, 1 m/s, random initial direction
MAC layer	B-MAC (preamble length: 100 ms)
Radio model	Half-duplex, bandwidth: 15 kB/s
Antenna model	Omnidirectional
Radio propagation model	Friis, 868 Mhz, pathloss: 2 (range: 4 m)
Modulation model	BPSK
Duration and number	100 s, simulated 20 times for every possible combination

Table 6: Simulation parameters used in our experiments.

1 to 50 for the mobile sensor) are randomly distributed in a 20x20 m regular square grid. An example of topology (with 100 fixed nodes) used in our experiment is depicted in Figure 20. All the sensors use the B-MAC protocol to access the medium, with a preamble length configured to 100 ms. The mobile sensor moves following a billiard mobility model (in which the node rebound against the network boundaries) at a speed of 1 m/s. The initial position and direction of the mobile sensor is randomly chosen for each simulation. The radio used on each sensor node is half-duplex (it cannot transmit and receive simultaneously) and implements a capture effect: if the radio is locked on a signal but receives a stronger one, it looses the first one and locks on the second one. The antenna operates in an omnidirectional fashion. A Binary Phase-Shift Keying (BPSK) digital modulation scheme is employed to modulate the phase of the simulated signal. Radio propagations are simulated with a log-distance pathloss propagation model. Propagations are however limited to a range of 4 m in order to better estimate the neighborhood of the mobile sensor during the experiment.

Our experiment focusing on the medium access, no routing protocol is being used on the nodes. Each sensor node thus sends its data packet in a broadcast fashion, and do not forward any of the received packets. Therefore the broadcast scheme does not cause any exponential traffic load in the network, which remains

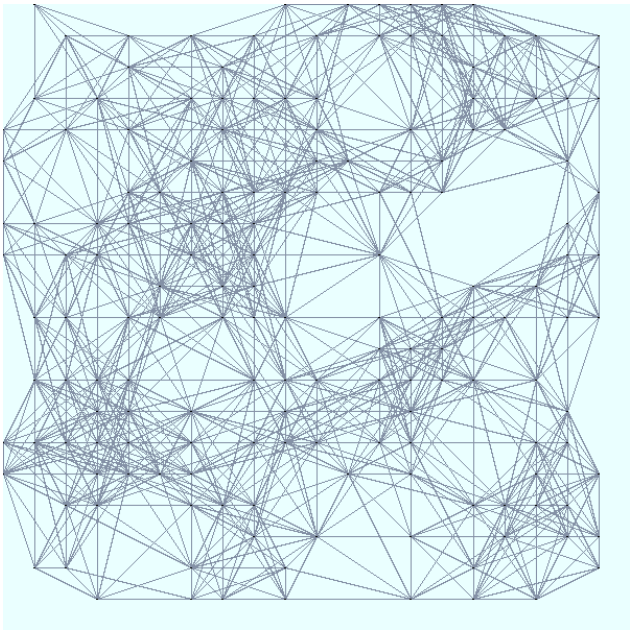


Figure 20: Example of topology and links with 100 sensor nodes.

uniform throughout the simulation. When computing the packet losses for the mobile sensor, we consider that a frame is lost when none of its neighbors received the broadcast packet it sent.

Every pair in *(number of fixed sensors, data sending period)* has been simulated 20 times. The results outlined in the next Section are an average of the overall data collected on the set of simulations. The 95% confidence interval displayed in our results denotes the reliability of our measures over these 20 trials.

3.2 RESULTS

We detail below the results obtained in terms of packet losses, medium access delay and duty cycle at the mobile node.

3.2.1 *Packet losses*

We have computed the packet losses of a mobile sensor, i.e. the percentage of packets sent by the mobile at the application layer and received by none of its neighbors at the end of the simulation. It is depicted according to the average number of neighbors in Figure 21. We notice a packet loss rate above 10% for B-MAC when the average number of neighbors is below 3.77 (which represents a maximum of 30 fixed nodes in the network). Such losses are also experienced when the neighbor density is above 30 (i.e with more than 250 fixed nodes in the simulation) and communications are frequent in the network (data sending period of 1 s on the fixed nodes).

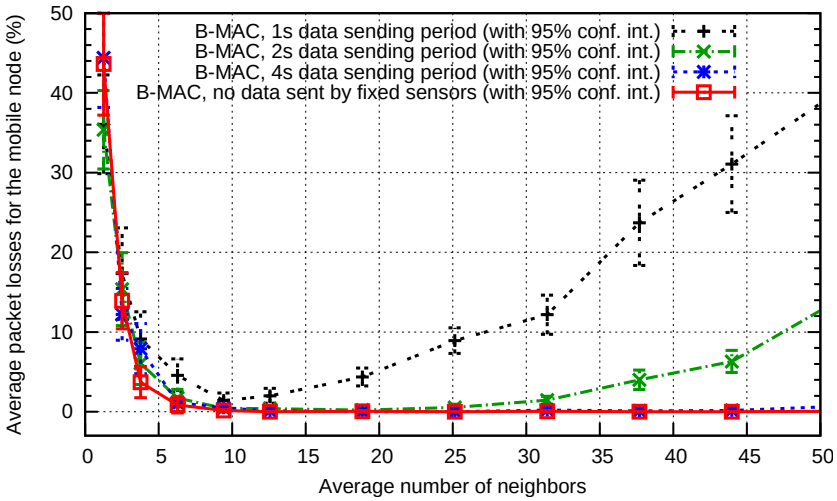


Figure 21: Average packet losses for a mobile sensor using B-MAC, according to its average number of neighbors.

Various reasons can explain why packets from the mobile sensor are not received by peers. The main ones are summarized in Table 7. In the first case, it turns out that the network density is not high enough to guarantee that the mobile sensor will be in the surrounding of other sensors when it emits its data: packets are obviously lost (column *No neighbor*). In the second case, we notice that most of the packets not received at the end of the simulation are actually still in the packet queue of the mobile node (column *In queue*). Table 7 also exposes other causes to the losses observed for B-MAC. A *switched off radio* while receiving a data packet illustrates a synchronization issue between the sender and the receiver. Most likely, the receiver did not capture

Number of fixed nodes	*Number of neighbors*	*Packet losses (%) and 95% conf. int.*	*Main reasons (% of the packet losses)*				
			No neighbor	*In queue*	*Packet error*	*Radio off*	*Packet not captured*
10	1.26	36.07 (± 6.19)	95.42	0.00	0.79	2.10	1.18
20	2.51	17.33 (± 5.73)	84.10	0.00	3.92	6.53	5.01
50	6.28	4.56 (± 2.06)	31.87	1.10	15.17	23.98	26.42
100	12.57	2.00 (± 0.92)	0.00	15.00	15.34	31.66	37.02
150	18.85	4.36 (± 1.12)	0.00	16.09	13.85	20.43	49.25
200	25.13	8.92 (± 1.60)	0.00	22.47	13.99	10.46	52.66
250	31.42	12.17 (± 2.45)	0.00	33.33	12.22	5.68	48.35
300	37.70	23.70 (± 5.35)	0.00	63.21	6.47	2.50	27.62
350	43.98	31.06 (± 6.06)	0.00	70.32	5.28	1.30	22.90
400	50.27	39.03 (± 6.42)	0.00	74.58	4.79	0.80	19.74

Table 7: Main reasons why packets from a mobile node are lost using B-MAC, when the data sending period is 1 s.

the preamble correctly (due to a collision or a packet error) and thus switched off its radio prematurely. A *data packet not captured* means that the radio has captured another packet (preamble or data) instead, certainly because the reception sensibility was better for that packet. A *packet error* happens for a data packet when it was captured correctly, but dropped upon reception because it contained errors caused by a collision. All of these three causes initially happen because the receiver node hears multiple transmitters at the same time. Although one of the role of the preamble is to reserve the medium, we do not use any mechanism such as RTS/CTS to reduce the hidden node problem.

3.2.2 *Medium access delay*

Figure 22 illustrates why B-MAC experiences filled up packet queues in a dense network. It depicts the average delay per packet for the mobile sensor to access the medium. As a reference, the case where fixed nodes do not send any data during the experiment gives us the lower bound of this delay. For B-MAC, it is equivalent to 106 ms ($\pm$ 0.12) in average, that is the sum of the backoff period (5 ms in average), the preamble and SYNC message (100 ms), and the time to sample the channel (1 ms in average). We notice that the medium access delay can exceed the data sending period of the mobile node at the application layer (1 s in our environment). The high level of competition on the channel is the cause of such delay. As a consequence, data packets at the mobile sensor are pushed in the communication queue faster than they are sent on the medium. According to the size of

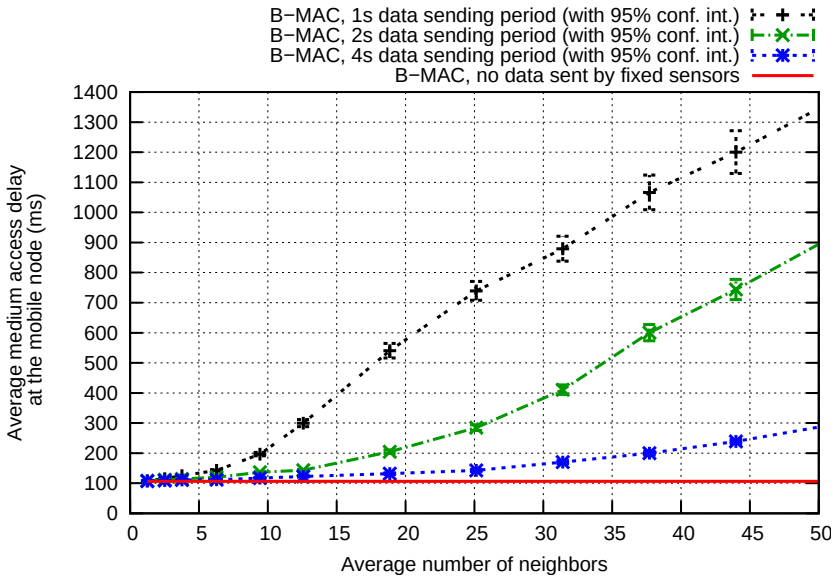


Figure 22: Average medium access delay for a mobile sensor using B-MAC, according to its average number of neighbors.

the queue, packets may be either dropped or greatly delayed. As an example from Table 7, our network with 400 sensors exhibits a packet loss rate of 39.03% (± 6.42) at the mobile node, in which nearly 75% corresponds to packets that are still in its queue at the end of the simulation.

3.2.3 *Duty cycle*

Another interesting aspect to measure is the radio duty cycle of the mobile sensor. It is depicted on Figure 23. As a reference, when no fixed node sends data on the medium, the duty cycle of the mobile node is in average 12.8% (± 0.13), which means that its radio is on during 128 ms every second. This is consistent with the fact the the mobile sensor sends a data packet every second. In case the competition on the channel is high, the 30% threshold is exceeded with B-MAC even in lowly dense networks. The main reasons are that the mobile node constantly has data packets in its queue, and needs to send a preamble for each of them. The mobile is thus constantly in receive mode (sampling the channel waiting for an opportunity to transmit its data) or in transmission mode (sending the preamble and data packet).

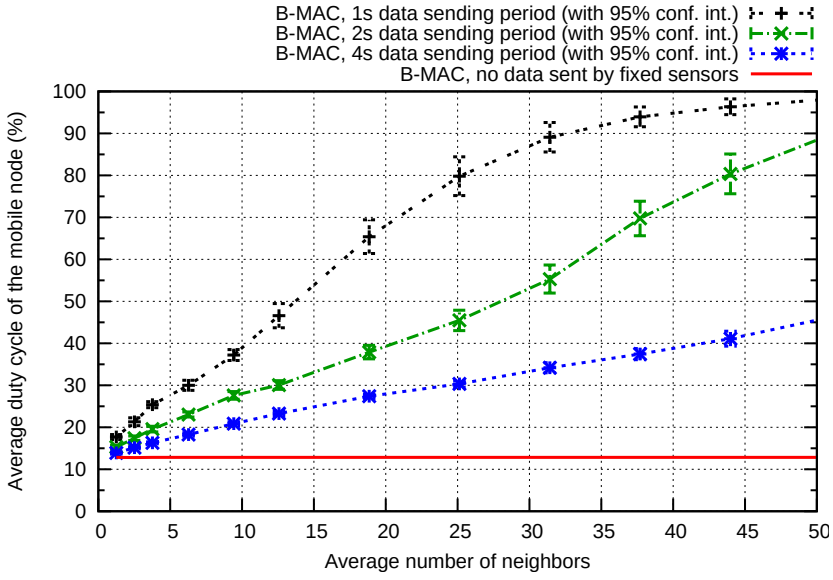


Figure 23: Average radio duty cycle for a mobile sensor using B-MAC, according to its average number of neighbors.

3.3 CONCLUSION

Preamble sampling protocols have appeared as a likely solution to deal with mobility at large scale, because no global medium reservation mechanism is required. However, the use of a preamble (which reduces the channel availability) has questioned us on the potential medium access issues that could appear in dense networks. In order to get a better understanding of these problems, we have experimented in this Chapter the operation of a simple, widely established protocol (B-MAC) in the context of a mobile sensor evolving in a network whose density and data traffic increase.

We have identified three main issues when operating such a preamble sampling protocol in a mobile environment. First, the synchronization between fixed and mobile sensor nodes can fail, hence provoking packet losses. Also, channel congestion issues increase the medium access delay of the mobile sensor node, which in turn can saturate its data packet queue. The battery of the mobile may sustain the mobility system in addition to the node. It would thus be interesting to reduce the duty cycle of the mobile sensor node. As an attempt to alleviate these issues, we propose the *Machiavel* protocol in the next Chapter.

4

PRELIMINARY CONTRIBUTION: MACHIAVEL

Contents

In order to alleviate the issues raised in Chapter 3 with B-MAC, we have specified the *Machiavel* protocol [56, 57] especially designed for mobile sensors. This first contribution focuses on a mobile node wishing to access the medium in a dense network mainly composed of fixed sensors. It extends the B-MAC protocol in order to make it more performant in a mobile environment. The principles behind Machiavel can be applied to other preamble sampling protocols: we will particularly expose in Chapter 5 how we have carried on this work with an optimized protocol (X-MAC) in order to support a high number of mobile sensors.

4.1 PROTOCOL OVERVIEW

In a nutshell, Machiavel allows mobile sensors to take possession of the medium initially owned by a fixed node. A small interval observed by fixed sensors between the preamble and the data gives the opportunity to a mobile node to send its data prior to the sender of the preamble. We will first describe how Machiavel operates at one hop in Section 4.1.1, followed by the multi-hop operations in Section 4.1.2.

4.1.1 *First-hop operations*

Machiavel is based on B-MAC and hence is a sampling protocol: the preamble followed by a short SYNC message sent by a fixed sensor allows the neighborhood to prepare for the reception of the trailing data. Machiavel makes the mobile sensors benefit from this synchronization step. It is important to note that each sensor must know whether it is fixed or mobile. This could be

achieved either with additional hardware (such as a GPS chip or a gyrometer), using localization algorithms [43], or by statically defining the role on each node.

Mobile node operation on an idle channel

When a mobile node wishes to emit data, it first samples the medium. If it does not detect any signal, it follows the standard procedure as defined by B-MAC and detailed in Figure 24: it sends a preamble, a SYNC frame and then the data. The data is sent in an unicast fashion if the mobile sensor operates a routing protocol, or in broadcast. In the former case, the choice of the next hop is left to the routing layer. Upon reception by the destination peer, the unicast data frame can optionally be acknowledged.

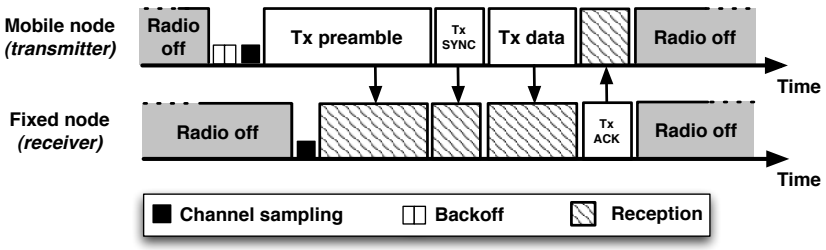


Figure 24: A mobile sensor wants to emit while the channel is free: it follows the standard B-MAC procedure.

Fixed node operation on an idle channel

Figure 25 illustrates the situation when a fixed node wants to transmit data on the medium. Fixed nodes behave in a very similar way as with B-MAC, but have to honor a delay (MIFS, Machiavel Inter-Frame Space) between the SYNC and the data frame. The value of the MIFS delay may vary according to the time a sensor node takes to sample the channel. Its value will be further discussed in Section 4.2. The mobile node will take the opportunity offered by this interval to take possession of the medium.

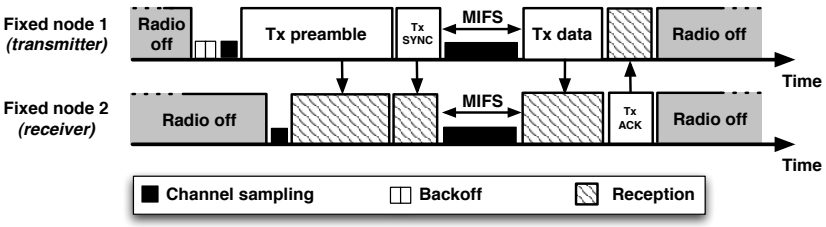


Figure 25: Fixed nodes send their data frames after the preamble and the SYNC, upon expiration of the MIFS delay.

Mobile and fixed nodes operations on an occupied channel

If the mobile node wants to send data but detects a preamble when sampling the channel, it operates as depicted in Figure 26. The mobile sensor is allowed to take possession of the medium during the MIFS delay, right after the end of the preamble and SYNC being sent by a fixed node.

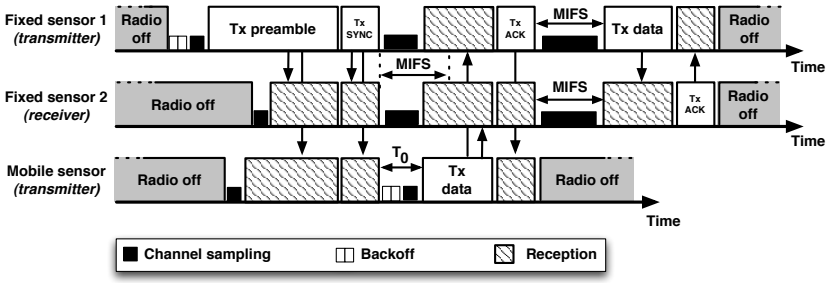


Figure 26: A mobile sensor wants to emit while the channel is occupied: it takes possession of the medium during the MIFS delay.

In order to minimize the risk of collisions with other sensors, the mobile node draws a random time T_0 between 0 and MIFS. Upon the expiration of this delay, it samples the medium again, and sends its data if the channel is still free. However, the next hop chosen by the routing layer and used as a destination for the data sent by the mobile node may not match the address of the sender of the preamble. Therefore, if the data is not a broadcast frame, the mobile node sets its destination to the source of the received SYNC message. The sensor that initially emitted the preamble, as well as all the other sensors located in the shared neighborhood of the mobile node, receive the data. The intended receiver can optionally acknowledge it if it was sent in unicast. Upon reception of the acknowledgment, the mobile sensor can switch off its radio if it does not participate in the routing. All of the other recipients know that more data will follow as long as the source address of the received data does not match the one of the SYNC message. Before being able to switch off their radio, they thus wait a delay at least equal to MIFS. During this period, other mobile nodes may send their own data following the same scheme as previously explained. If no other mobile emits data upon expiration of the MIFS delay, the fixed sensor which initially owned the medium can send its data. On reception, all of the recipients can switch off their radio because the source address of the data frame matches the one of the SYNC message.

The performances of a fixed node may be impacted if too many mobile sensors take possession of the medium that it owns (we will further evaluate this aspect in Section 4.2). For that purpose, the fixed sensor can restrict how many mobile nodes could consecutively take possession of the channel by sending

its data immediately after the one of the mobile node, without waiting for the MIFS delay.

Note that mobile nodes operating Machiavel cannot steal the medium to each others: as detailed in Figure 24, a mobile node sending a preamble does not need to observe the MIFS delay.

4.1.2 *Multi-hop operations*

The first-hop operations described in the previous Section would allow to reduce synchronization and congestion issues at the mobile node. However, in a multi-hop topology, the data emitted by a mobile node and forwarded by the fixed nodes toward the sink may still suffer from congestions. According to the number of hops, the gain obtained at the first hop may thus become negligible. Therefore, a bit in the Machiavel MAC header indicates whether the data initially comes from a mobile node. When a fixed node has to forward such data, it stores it in a priority packet queue. A fixed node always checks first in this queue when it is ready to forward data. When trying to access the medium to send such priority packet, the fixed node is allowed to take possession of the channel in the same way as a mobile node. As a result, packets initially emitted by a mobile node can be forwarded faster to the sink.

However, this scheme does not guarantee that the next routing hop will be synchronized and ready to receive data. As detailed in Figure 27, the next routing hop may not be in range of the node that initially sent the preamble (the *preamble sender*), and thus is not synchronized with the *data sender* node upon transmission of its data frame. For that reason, fixed nodes may only take possession of the medium when it is owned by a node located in the shared neighborhood with its next routing hop, or when it is owned by the next routing hop itself.

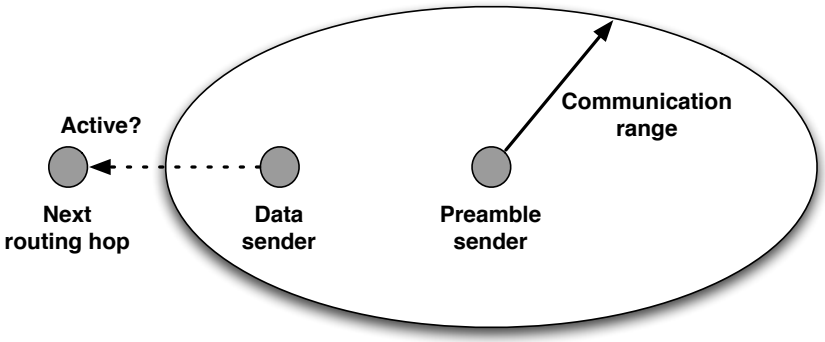


Figure 27: Fixed nodes can take possession of the medium only if their next routing hop can hear the preamble sender.

4.2 EVALUATION OF OUR CONTRIBUTION

We have implemented Machiavel for WSNet. The simulation parameters employed to evaluate our contribution are the same as presented in Section 3.1 (Table 6). We have only replaced the B-MAC model on both the mobile and fixed sensor nodes with Machiavel. We have used a value of 1 ms for the MIFS delay. This duration allows fixed sensors to switch from transmission to reception modes, sample the medium and analyze that sample in order to verify whether a mobile sensor took possession of the channel (The B-MAC authors indicate in [77] that the whole operation can take around 0.7 ms on a CC1000 radio).

In the below Sections, we first evaluate the benefits of Machiavel from the mobile node point of view. We then exhibit its cost on the fixed sensors. Finally, we analyze its performances in a multi-hop scenario. Machiavel aims at improving B-MAC in a mobile environment. As a matter of comparison, the Figures presented hereinafter thus expose one significant result from the B-MAC evaluation (Chapter 3). The 95% confidence interval displayed in our results confirms the reliability of our measures.

4.2.1 *Benefits for the mobile sensors*

In dense networks, Machiavel significantly reduces the packet losses at the mobile node compared to B-MAC, as detailed on Figure 28. It is close to zero, even when communications are frequent. This means that a large majority of the packets sent by the mobile sensor has been received by its peer. However,

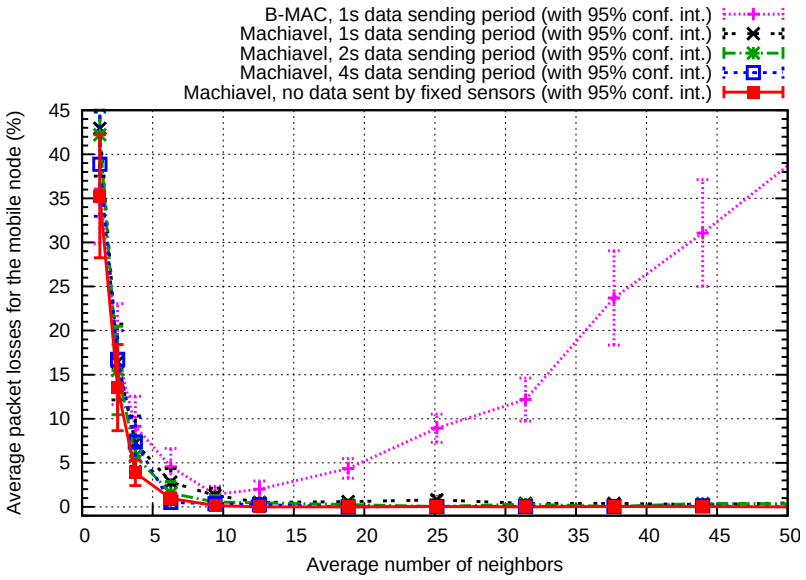


Figure 28: Average packet losses for a mobile sensor using Machiavel, according to its average number of neighbors.

Number of fixed nodes	*Number of neighbors*	*Packet losses (%) and 95% conf. int.*	*Main reasons (% of the packet losses)*				
			No neighbor	*In queue*	*Packet error*	*Radio off*	*Packet not captured*
10	1.26	42.94 (± 5.42)	95.45	0	0.23	2.57	1.75
20	2.51	16.43 (± 4.30)	82.32	0	2.21	8.10	6.88
50	6.28	2.81 (± 1.54)	37.50	0	5.21	32.29	22.92
100	12.57	0.50 (± 0.42)	0	0	19.64	37.50	42.86
150	18.85	0.60 (± 0.35)	0	16.67	19.08	18.07	40.16
200	25.13	0.80 (± 0.49)	0	43.75	12.18	10.44	28.41
250	31.42	0.40 (± 0.28)	0	75.00	4.65	1.74	13.95
300	37.70	0.40 (± 0.24)	0	62.50	6.41	1.97	22.20
350	43.98	0.25 (± 0.26)	0	100.00	0	0	0
400	50.27	0.45 (± 0.32)	0	100.00	0	0	0

Table 8: Main reasons why packets from a mobile node are lost using Machiavel, when the data sending period is 1 s.

when the neighbor density is low, the mobile sensor operating Machiavel experiences similar packet losses than with B-MAC. This can be explained by the same reasons as previously: packets are lost because no neighbor is in the surrounding of the mobile sensor, as confirmed by Table 8. Although Table 8 still reveals losses in dense networks due to congestions or synchronization issues, they are negligible if we confront them to the total of lost packets.

Figure 29 illustrates why Machiavel performs better than B-MAC in a dense neighborhood. It depicts the average delay per packet

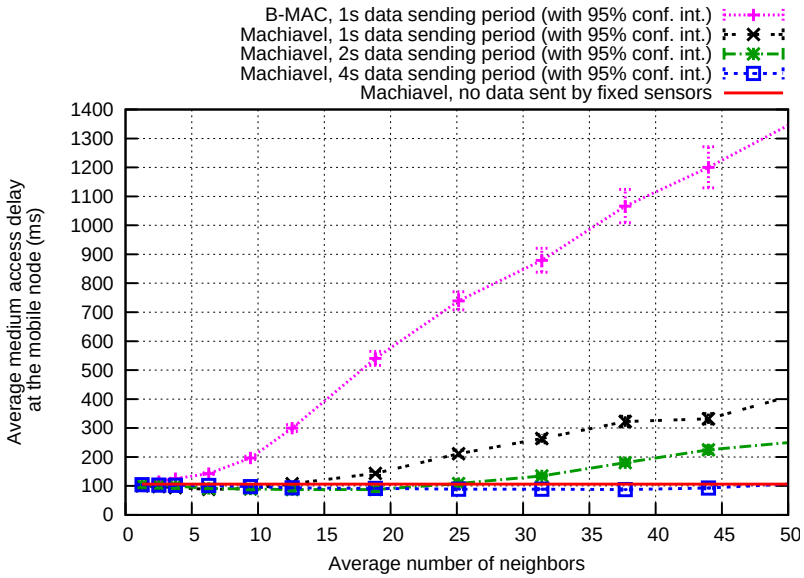


Figure 29: Average medium access delay for a mobile sensor using Machiavel, according to its average number of neighbors.

experienced by the mobile node to access the medium. With Machiavel, the medium access delay is greatly reduced. By allowing the mobile sensor to take possession of the medium, Machiavel gives it the opportunity to emit its data even though the level of competition on the channel is high. The mobile may succeed in sending its own data right after the first preamble sampled, which results in delays smaller than the lower bound calculated for B-MAC or Machiavel when no data is sent by fixed sensors (106 ms, see Section 3.2). However, this is not always the case, as we can also observe that delays may be longer than the Preamble Length. If the medium is occupied during the MIFS delay (e.g. when a collision occurs between two fixed nodes), the mobile node does not send its data and waits for a later opportunity. In any case, this delay always remains lower than the data sending period of the mobile node (1 s). Its queue is thus never filled to capacity.

The radio duty cycle of the mobile sensor operating Machiavel is depicted on Figure 30. As a reference, when no fixed node sends data on the medium, the duty cycle of a mobile node operating Machiavel is the same as the one computed with B-MAC (12.8%, see Section 3.2.3). This is obvious since the mobile node follows the B-MAC protocol when the channel is idle. However, we can observe a reduction of almost 30 points compared to the duty cycle obtained for B-MAC when the competition is high in the most dense networks.

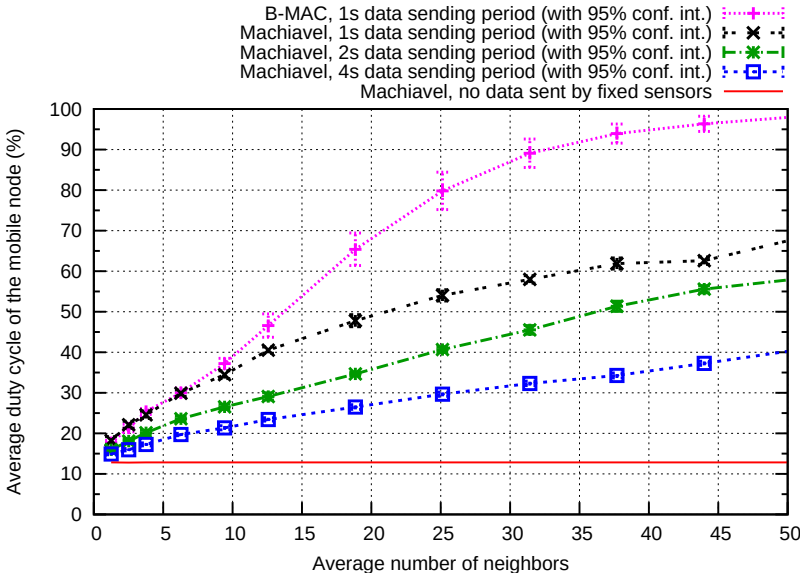


Figure 30: Average radio duty cycle for a mobile sensor using Machiavel, according to its average number of neighbors.

4.2.2 *Cost for the fixed sensors*

Yet, it is obvious that Machiavel has a cost for the fixed nodes and increases their duty cycle. In case N mobile sensors take successively possession of the medium that was initially owned by the same fixed node, we compute the time T_m needed by this fixed sensor to send its own data. It is estimated according to the backoff T_o, the channel sampling time T_e, the time T_p required to send the preamble and the SYNC message, and the time T_d to send the data. T_m can be computed with the following formula:

$$\begin{aligned} T_m &= T_o + T_e + T_p + MIFS * (N+1) + T_d * (N+1) \\ &= T_o + T_e + T_p + (MIFS + T_d) * (N+1) \end{aligned}$$

The corresponding time for a fixed sensor using B-MAC would be the time T_b such as: $T_b = T_o + T_e + T_p + T_d$. The additional cost C_m produced by Machiavel on the fixed nodes compared to B-MAC is thus:

$$C_m = \frac{T_m - T_b}{T_b} = \frac{MIFS * (N+1) + T_d * N}{T_o + T_e + T_p + T_d}$$

When no mobile sensors take possession of the medium (i.e. $N = 0$), we obtain:

$$C_m = \frac{MIFS}{T_o + T_e + T_p + T_d}$$

In that special case, by using the values used in our experimentations (T_o = 5 ms in average, T_e = 1 ms, T_p = 100 ms, MIFS = 1 ms, T_d = 1.2 ms), we obtain $C_m = 0.93\%$. Compared to B-MAC, this overhead is equivalent to the MIFS delay (1 ms). We have also evaluated T_m by simulation. Both the simulation and analytical results are depicted in Figure 31, according to the number of mobile nodes that successively take possession of the medium. We can note that the simulation results match the analytical ones.

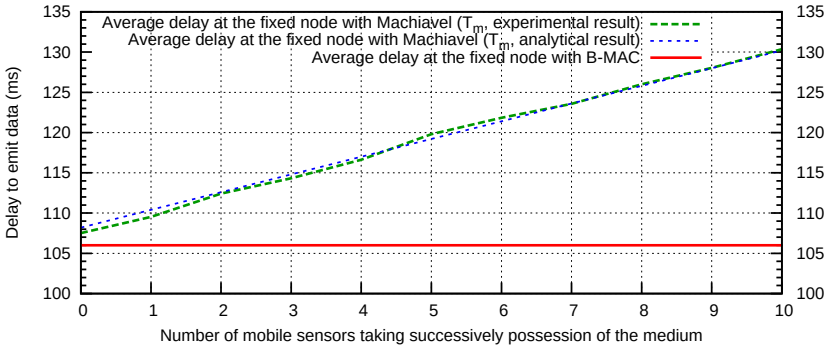


Figure 31: Average delay to emit data for a fixed node, according to the number of mobile sensors that successively take possession of the medium.

With $N = 1$, we obtain $C_m = 2.98\%$. If $N = 10$, the delay to emit data for the fixed sensor reaches 130 ms, which represents a 22.6% overhead compared to B-MAC. This highlights the need to limit the number of time the medium can be possessed successively by mobile sensors, as we discussed already in Section 4.1.1. For example, such limit could be dynamically adjusted by the fixed node according to its remaining battery. In return, we can raise that mobile nodes do not need to send a preamble anymore. For example, data from all the ten mobiles can be sent within 130 ms. With B-MAC, it would take more than a second (as each would need to capture the medium, send a preamble and its data).

4.2.3 *Multi-hop operations*

In order to evaluate how Machiavel behaves in a multi-hop scenario, we have added static routing toward the sink in our simulation environment. We have evaluated the end-to-end delay between a mobile sensor and a sink according to the number of hops between both, when using B-MAC or Machiavel. At the beginning of the simulation, the packet queues of all the sensor nodes are empty. We compute the average delay for the first packet sent by the mobile node over 20 simulation trials. Results are depicted in Figure 32.

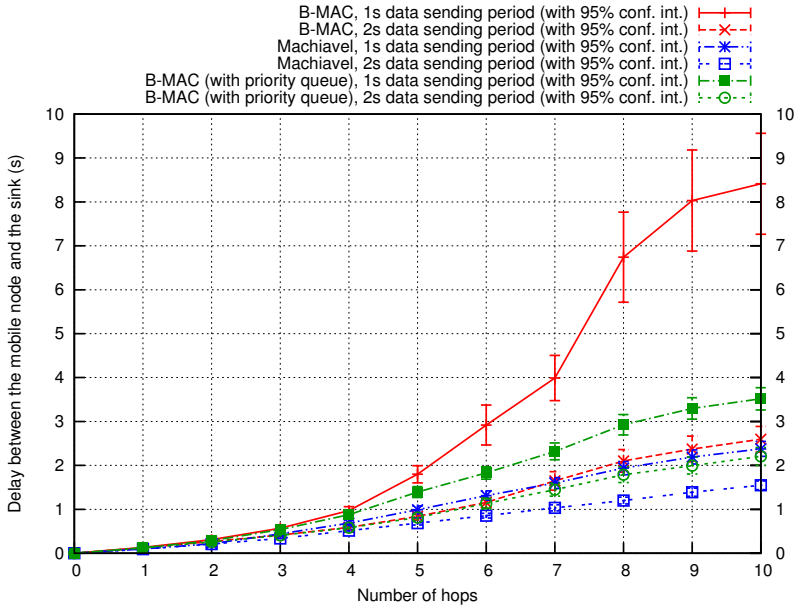


Figure 32: Delay from the mobile node to the sink in a multi-hop scenario, according to the number of hops.

We can observe a sudden increase after the seventh hop for B-MAC with a 1 s data sending period. This is due to the funneling effect which causes the packet queues of the nodes to be filled up along with the number of hops. The use of a priority queue that

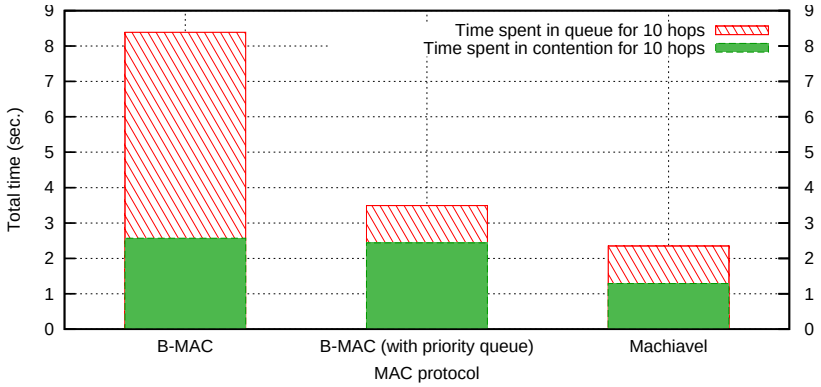


Figure 33: Average time spent in contention and in queue over 10 hops for B-MAC and Machiavel.

we raised in Section 4.1.2 can alleviate this problem. As this idea is not new, we have also implemented this mechanism in B-MAC in order to better compare the gain offered by our contribution. Still, Machiavel performs better than B-MAC by allowing the fixed nodes to take possession of the medium when forwarding data originating from a mobile node.

In order to illustrate how this plays a role in the overall latency, we represent in Figure 33 the average time spent in queue and in contention for a packet traveling through 10 hops. We can notice that Machiavel greatly decreases the time spent in contention. The priority queue in combination with our protocol reduces the overall delay by a factor of 3.6 compared to B-MAC, and 1.5 compared to B-MAC with a priority queue.

4.3 CONCLUSION

With this first contribution, we have particularly focused on the synchronization failures between fixed and mobile nodes, as well as the channel congestion problems in dense networks operating a preamble sampling protocol. In order to alleviate these issues, we have proposed the Machiavel protocol. Unlike standard sampling protocols, Machiavel guarantees a mobile node that its neighbors are synchronized when emitting data. Machiavel also reduces the delay to access the medium, hence avoids the mobile to saturate its packet queue. When operating over several hops, the combination of the medium borrowing with a priority queue allows to report data faster to the sink. An evaluation of our proposal has demonstrated significant reduction in packet losses and end-to-end delays in dense networks compared to the B-MAC protocol on which it is based.

As Machiavel relies on an infrastructure of fixed sensors, we need to investigate further its behavior when the ratio of mobile to fixed nodes increases in the network. Also, we would like to

extend our contribution to the communications between mobile sensors. With the recent and performant preamble sampling protocols that were proposed (such as X-MAC [17]), we would like to demonstrate that the principles behind Machiavel can be applied to other preamble sampling schemes as well, in order to further improve their performances in a mobile environment. In the next Chapter, we will try to address these points with the X-Machiavel protocol.

5

THE X-MACHIAVEL PROTOCOL

Contents

In Chapter 4, we have exposed the Machiavel scheme, which extends B-MAC in order to guarantee that a mobile node and its peer are correctly synchronized during the data transmission. In a nutshell, Machiavel allows mobile sensors to take possession of the medium initially owned by a fixed node. A small interval (MIFS) observed by fixed sensors between the preamble and the data gives the opportunity to a mobile node to send its data prior to the sender of the preamble. Even though Machiavel reduces the medium access delay of the mobile nodes with little overhead for the fixed sensors, it was not designed to support large number of mobiles and focused on convergecast communications only. Furthermore, several protocols have proven to be more performant than B-MAC on which Machiavel is based. For example, X-MAC [17] optimizes the LPL mechanism by using a strobed preamble and allowing receivers to acknowledge it, which triggers the data transmission (see Figure 11 in Chapter 1). X-MAC mitigates both overhearing on non-recipient nodes and long preamble transmissions on senders.

In this chapter, we present the *X-Machiavel* protocol [54]. It extends Machiavel by applying its principles (taking possession of the medium owned by a fixed node) on the X-MAC protocol, and brings several novelties. It takes benefit of the strobed preamble mechanism of X-MAC to integrate the mobile nodes in the communication schedule. Mobile-to-mobile communications are supported in networks with great numbers of moving sensors.

Furthermore, mobile nodes operating X-Machiavel do not need to maintain a list of neighbors, which would be complex to achieve in such dynamic topologies.

X-Machiavel makes the assumption that the data originating from a mobile node are more important than from a fixed one. This is motivated by the fact the a mobile sensor may not be in the vicinity of the perceived event forever, and may not have the capacity to store much data in queue. Also, mobile nodes implementing X-Machiavel do not participate in forwarding operations, as routing in a mobile environment is hard to achieve. Instead, an infrastructure of fixed nodes takes care of routing the data packets between end nodes. As X-Machiavel defines different operations according to the role of the node (mobile or fixed), we also assume that each node knows whether it is moving or not. As explained in Section 4.1.1, this could be achieved either with additional hardware (such as a GPS chip or a gyrometer), using localization algorithms [43], or by statically defining the role on each node.

5.1 PROTOCOL OVERVIEW

The X-Machiavel protocol is designed to reduce the medium access delay of the mobile nodes in both lowly and highly contended networks. It also guarantees in both cases that the mobile and its next hop are synchronized during the transmission of the data. After a brief overview of the X-Machiavel header and the terminology in Section 5.1.1, we explain the operation of the protocol when the channel is free in Section 5.1.2, while its behavior on a contented medium is detailed in Section 5.1.3. Section 5.1.4 describes specific multi-hop operations.

5.1.1 *X-Machiavel header*

The X-Machiavel header is depicted in Figure 34. It is included in the strobed preamble frames, data packets and acknowledgment messages. The source (*src*) field is set to the ID of the node that sends or forwards the packet. On fixed nodes, the destination (*dst*) field is set to the ID of the next hop, as provided by the routing algorithm. However, we make the assumption that mobile sensors using X-Machiavel do not have any information about their neighbors. Routing in WSNs is a vast topic still being investigated [3]. Maintaining valid routes on mobile sensors is an awkward issue and may require complex algorithms, expensive hardware (such as a GPS chip) or frequent control messages overhead. For these reasons, mobile nodes using X-Machiavel always initially set the MAC destination address to the ID of their final destination (hereinafter called the *remote peer*). As we will explain

in the next Sections, mobile sensors will be able to replace this address with their real next hop when a data packet is to be sent.

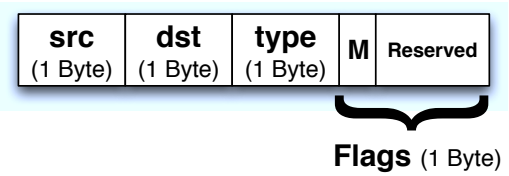


Figure 34: The X-Machiavel header.

The *type* field defines whether the packet is a preamble frame (type *Po*, *P1*, or *P2*), a data packet (*DATA*), an acknowledgment for a preamble (*PKo* or *PK1*) or an acknowledgment for a data packet (*ACK)*. Strobed preambles can be of three different types: *Po* preamble frames are sent by mobile nodes only and notify other nodes that the channel cannot be stolen (see Section 5.1.3). *P1* and *P2* preamble frames are sent by fixed nodes and respectively inform other sensors whether the channel can be subtilized (*P1*) or not (*P2*).

Two types of preamble acknowledgments can be used: *PKo* is sent by fixed sensors to acknowledge a *Po* preamble frame that was not initially intended to them. It notifies the mobile node that the fixed sensor can take care of its data. The *PK1* acknowledgment message is used in other cases, i.e. when nodes acknowledge a preamble that is destined to them.

Flags are used to notify the type of the packet transiting in the network. Mobile sensors always set the *M* flag for data packets, which remains even when a fixed node forwards such packets. Before forwarding a data that has the *M* flag set, a fixed node always prevents the channel from being stolen by using a *P2* preamble. The *Reserved* space may be used for other flags in future evolutions of the protocol.

The use of each type and flags will be further explained along with the protocol operations in the next Sections.

5.1.2 *Operation on an idle channel*

A mobile sensor wishing to send a data packet while the medium is idle starts sending a strobed preamble of type *Po*. The destination field of each preamble frame is set to the ID of the remote peer. If that peer happens to be in the neighborhood of the mobile sensor and hears the preamble, the principles of the X-MAC protocol applies (Figure 35): the peer sends a *PK1* acknowledgment without delays in order to claim the data to the mobile node.

However, in large deployments, it is likely that the remote peer is not in the vicinity of the mobile node. For that reason, a fixed node (hereinafter called a *forwarder*) located in the neigh-

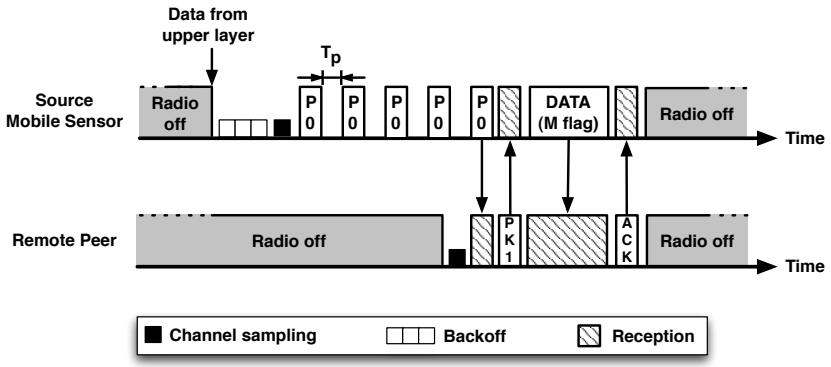


Figure 35: Mobile-to-peer communications use X-MAC (with specific header types) when the remote peer acknowledges the preamble.

borhood and which hears the *Po* preamble frame sent by the mobile sensor, can acknowledge it with a *PKo* acknowledgment (Figure 36). Note that only the fixed sensors are allowed to act as forwarders. Mobile nodes do not participate in the routing of the data frames. In order to avoid a collision between multiple potential forwarders, we define a very simple backoff mechanism. Forwarders draw a random delay T_w such as:

$$T_w = \mathrm{random}(\frac{T_p}{2}, T_p)$$

with T_p being the interval between two preamble frames. The use of such backoff also grants priority to the remote peer in case it would receive the preamble at the same time as a forwarder. Once the T_w delay expires, the forwarder performs a carrier sense and sends its *PKo* acknowledgment if the channel is still free. The forwarder may notice that a collision has occurred if it still receives *Po* preamble frames from the mobile sensor. In such case, it can try to send a *PKo* acknowledgment again.

The mobile node receiving the *PKo* acknowledgment sets the destination ID of its data packet to the source address of the received message: it explicitly destines the data to the forwarder that requests it. Upon reception of the data, the forwarder acknowledges it.

The data from the mobile sensor may now be routed through multiple fixed nodes until it reaches its final destination. Such destination could be a fixed sink in the case of a convergecast communication model, or another mobile node if we consider that fixed nodes have the appropriate routing algorithm. In both cases, data with the *M* flag set is transported hop-by-hop toward its final destination using *P2* preambles. In other words, the channel cannot be stolen (see Section 5.1.3) by potential mobile nodes when data originally coming from a mobile sensor is transiting

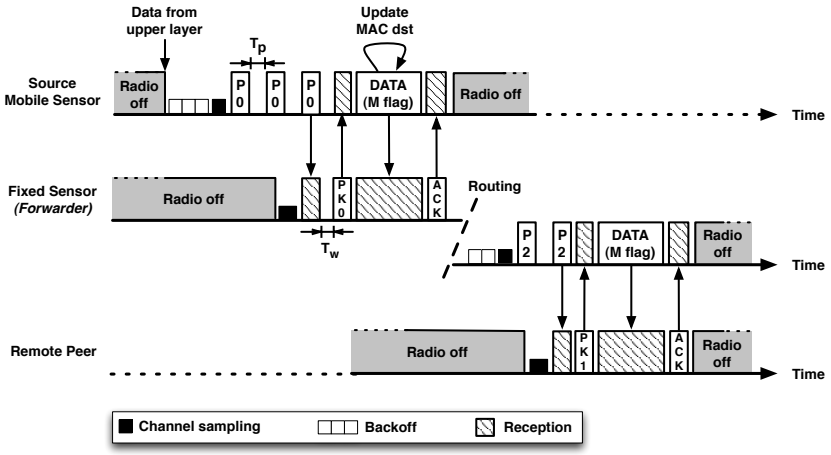


Figure 36: Mobile-to-peer communications when a forwarder acknowledges the preamble frame before the remote peer.

over the network. Furthermore, the use of *P2* preamble frames prevents potential forwarders from claiming the data.

Indeed, this scheme can add additional hops between the mobile node and its remote peer, in case the peer was actually located in the vicinity but woke up after a forwarder that has already claimed the data of the mobile sensor. We will further analyze in Section 5.2 whether such behavior represents a significant drawback.

5.1.3 *Operation on an occupied channel*

A mobile node traveling across a congested zone may suffer from long medium access delays. However, the node will not be in the vicinity of the perceived event forever, and may not have the capacity to store much data in queue. It can thus be necessary to integrate the mobile node in the neighborhood schedule, without aggravating the congestions in the area.

On an occupied channel, the mobile node can take the opportunity of the interval between strobed preamble frames to send its data packet to a forwarder (Figure 37). When the mobile sensor has data to send, it listens to the medium. If it overhears a *P1* preamble from a neighboring fixed node, and that preamble is not destined to itself, it can try to send its data to that forwarder between two consecutive preamble frames. For that purpose, it waits for a delay equivalent to T_w (to avoid potential collisions with other mobile sensors or with the preamble acknowledgment from the real destination of the preamble), performs a carrier sense and sends its data to the forwarder. Before transmitting the data, the mobile node sets the destination address to the source address of the received preamble frame.

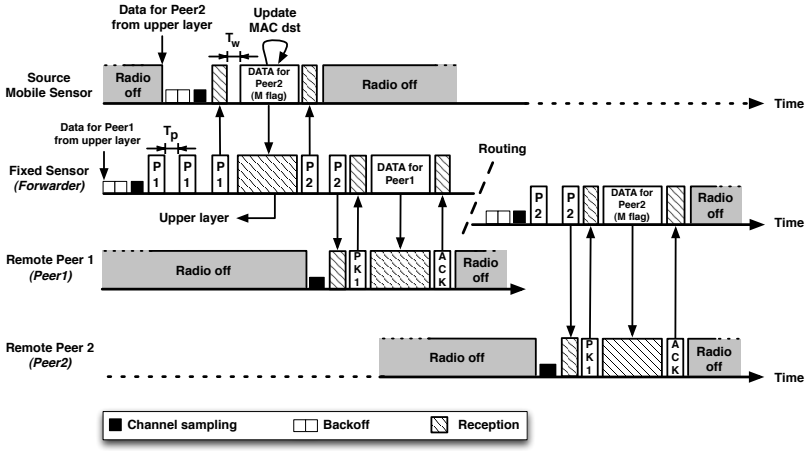


Figure 37: Mobile-to-peer communications when the channel is occupied by a fixed node.

The data received by the forwarder is sent to the upper layer for a later processing. The forwarder still has to send its original data to its peer node (noted as *Peer1* in Figure 37). In order to prevent another mobile node from capturing the channel again, the forwarder sends its next preamble frames with a *P2* type. By overhearing such a *P2* preamble from the forwarder, the mobile sensor knows that its data was correctly received. If not, it can try again later upon reception of a *P1* preamble frame.

Once the *P2* preamble is received by Peer1, it can initiate the data transmission as usual, i.e. by sending a *PK1* acknowledgment to the fixed node. Note that Peer1 may instead wake up during the mobile node transmission, sample the medium and go back to sleep because it does not receive any preamble during that interval. For that reason, the preamble sender (here, the forwarder) resets its Preamble Length counter before sending *P2* preamble frames in order to ensure that the destination will be able to receive it.

The data from the mobile node that was received by the forwarder can later be sent toward its final destination (*Peer2* in Figure 37). The operations are strictly the same as when the forwarder received data from a mobile node in the idle channel case (Section 5.1.2): it is transported hop-by-hop toward its final destination using *P2* preamble frames.

5.1.4 *Multi-hop operations*

The first-hop operations previously described would mitigate synchronization and congestion issues at the mobile node. However, in a multi-hop topology, the data emitted by a mobile sensor and forwarded by the fixed nodes toward the final destination may still be affected by congestions. According to the number of hops,

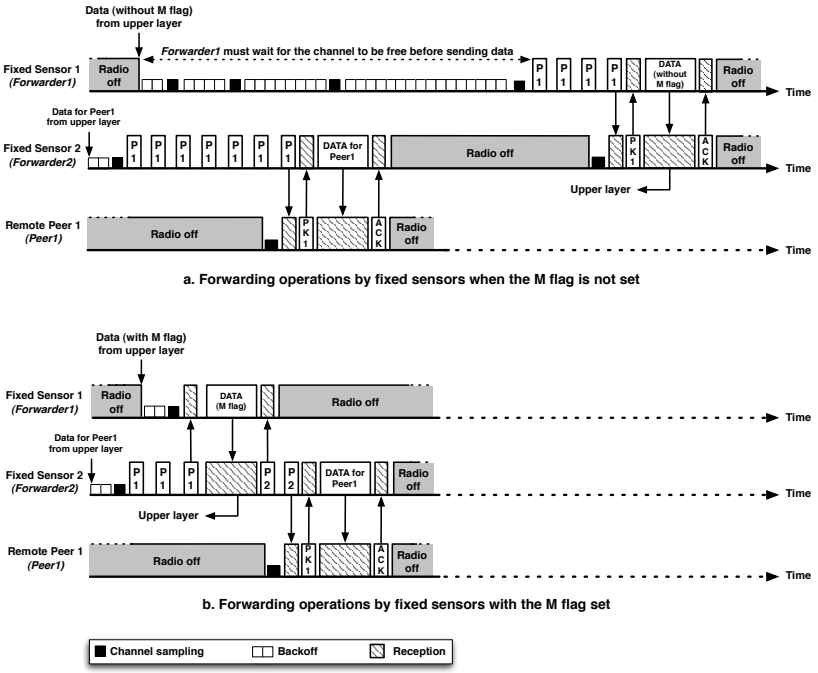


Figure 38: Multi-hop operations can be improved by allowing the fixed sensors to take possession of the medium when forwarding data from a mobile node.

the gain obtained at the first hop may thus become negligible. For example, Figure 38.a exhibits the usual forwarding operations of the fixed nodes, when the channel is congested. *Forwarder2* uses the channel to send data to *Peer1*. In such case, *Forwarder1* has to wait for the channel to be free if it also wishes to emit data. This results in multiple congestion backoffs at *Forwarder1*.

In case the destination of the data is another mobile sensor, it would be interesting to reduce the end-to-end delay because the routing information for moving destinations will fluctuate over time on the forwarding nodes. With X-Machiavel, fixed nodes can forward data originally emitted by a mobile sensor (i.e. with the *M* flag set) by using the same mechanism as described in Section 5.1.3. Figure 38.b shows that a significant amount of time can be saved. As a result, packets initially emitted by a mobile node can be forwarded faster to their respective destinations. However, as the fixed nodes operate a routing protocol and have a specific next hop to observe, they can only employ this mechanism when the source of the overheard preamble is equal to the layer-2 destination of the forwarded data.

5.2 EVALUATION OF OUR CONTRIBUTION

In order to evaluate our contribution, we have performed a set of simulations in a highly mobile environment. We will first

overview our simulation setup in Section 5.2.1. We have chosen to compare X-Machiavel with X-MAC in order to better estimate the benefits of our contribution with respect to the protocol on which it is based. In addition, X-MAC is one of the most popular preamble sampling protocol in WSN. We would like to show to what extent it is suitable to highly mobile scenarios. For that purpose, an evaluation of both protocols in terms of packet losses, medium access delay and energy consumption will be presented from Section 5.2.2.

5.2.1 *Simulation environment*

The evaluation of X-Machiavel was performed using the WSNet software. This simulator especially designed for WSNs has been previously described in Chapter 3 (Section 3.1). Table 9 details the WSNet models used in our simulations. 16 to 96 mobile sensors are operating around a fixed infrastructure of 16 nodes spread on a relatively large environment (10x10 m). This constitutes an increase from 1 to 6 in the proportion of mobile nodes compared to the fixed ones. This ratio is later referred to as R_n. Fixed nodes are positioned 2 meters from each others as a regular square grid, in a way that the whole simulation perimeter is covered at the radio level. Mobile nodes are initially distributed in a random fashion over the simulation grid, this placement being renewed for each simulation trial. The simulation starts with a warm-up period of 100 s, in which only the mobility model operates in order to further contribute to the random distribution of the mobile nodes. Then, each fixed and mobile node sends one data packet every 10 s (at the application layer, which randomly starts within a window of 10 s) during 1 hour, whose destination is randomly chosen among the whole set of sensors. The simulation ends with a cool-down period of 300 s, where no packet is sent by the application layer. This gives the opportunity to the sensor nodes to route their queued packets toward their destinations before the simulation completes.

In order to evaluate the improvements offered by X-Machiavel, we compare it to X-MAC. We set the Preamble Length and Sampling Period to 100 ms for both protocols. This value provides a good tradeoff between energy consumption and latency [77], and is usually the default one in reference implementations (e.g. on TinyOS [25]). For X-Machiavel, we have set T_p (the interval between two preamble frames, see Section 5.1.2) equals to 2 ms. This ensures that the nodes have enough time to switch from the reception to the transmission modes (and conversely) as well as receive potential acknowledgment frames. The energy model is configured from the consumption of a Chipcon CC1100 chipset [94]. As the radio sleep current is 400 nA (which is about

Simulation parameter	*Value*
Topology	Regular square grid (10x10 m) of 16 fixed sensors and 16 to 96 mobile sensors
Data collection scheme	Time-driven (10 s), peer-to-peer
Data packet/payload size	18 Bytes / 4 Bytes
Mobility model	Billiard, 1 m/s, random initial direction
Routing model	Fixed nodes: random geographic routing
MAC model	X-Machiavel or X-MAC with a preamble of 100 ms
Radio model	Half-duplex, bandwidth: 15 kB/s
Antenna model	Omnidirectional
Radio propagation model	Friis, 868 Mhz, pathloss: 2 (range: 3 m)
Modulation model	BPSK
Energy model With a 3 V battery	Idle: 1.6 mA, Rx: 15 mA, Tx: 16.9 mA, Radio init: 8.2 mA Idle: 4.8 mW, Rx: 45 mW, Tx: 50.7 mW, Radio init: 24.6 mW
Simulation setup	1 hour, simulated 20 times

Table 9: Simulation parameters used in our experiments.

40 000 times less than the reception current), we consider it as negligible in our simulations. The energy consumption results presented in the following Sections make the assumption that a 3 V battery is used (e.g. two 1.5 V AA cells, as used on TelosB or Mica2 motes [24]).

As we only focus on MAC optimizations, we assume an ideal routing protocol on the fixed nodes. The model used is a random geographic routing algorithm. A fixed node forwarding a packet selects its next hop randomly among the set of nodes that are strictly nearer from the destination than itself. Such list could be built for example using the MLS [37] algorithm, which is especially designed for routing in a mobile environment using location information.

As explained in Section 5.1, mobile sensors operating X-Machiavel do not participate in routing operations. They are only sender or receiver of the data. With X-Machiavel, mobile nodes set their the next hop at the effective time of data transmission, which significantly eases the routing management. Hence, no routing model is used on these nodes. However, mobile nodes running X-MAC have to implement a routing protocol in order to determine the next hop to which transmit their data. We have therefore performed our X-MAC experiments with two different routing algorithms. The first one is the same ideal geographic routing

algorithm as operated on the fixed sensors. We will refer to this case as X-MAC$_{mngeo}$. The second is a simple layer-2 broadcast mechanism. We will refer to it as X-MAC$_{mnbcast}$. It is important to note that with X-MAC$_{mnbcast}$, the strobed preamble transmitted by mobile nodes is also sent in a broadcast fashion, hence cannot be acknowledged earlier by peer nodes. We will thus be able to position our contribution with respect to X-MAC$_{mngeo}$ and X-MAC$_{mnbcast}$.

Other models (mobility, modulation, propagation, radio and antenna) have been detailed in Chapter 3 (Section 3.1). Each possible configuration of MAC layers and number of mobile sensors has been simulated 20 times with a different distribution of the data destinations, which corresponds to a total of 660 simulation trials. The results outlined in the following Sections are an average of the overall data collected on the set of simulations. The 95% confidence interval exposed in our results denotes the reliability of our measures.

5.2.2 *Losses at the application layer*

The average data packet losses at the application layer according to the ratio of mobile to fixed nodes (R_n) is depicted in Figure 39.

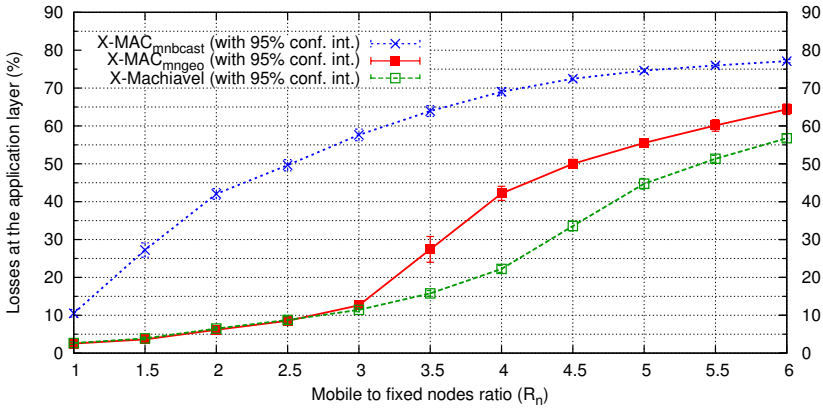


Figure 39: Average losses at the application layer according to R_n.

X-MAC$_{mngeo}$ and X-Machiavel achieves roughly the same delivery ratio up to $R_n = 3$ (48 mobile nodes). At that point, losses reach respectively 12.6% and 11.4% of the total of packets sent at the application layer. X-Machiavel manages to mitigate the losses when R_n further increases in the network. We will explain how this is achieved by analyzing the losses at the MAC layer in the next Section. With 64 mobile nodes ($R_n = 4$), 42.2% of the packets are lost when X-MAC$_{mngeo}$ is operated, while X-Machiavel can divide the losses by a factor 2 (22.2%). From $R_n = 5$ (80 mobile nodes), almost 1 packet over 2 is lost during the experiment for both protocols, which draws a limit in the average proportion of

mobile nodes that the network can handle. In the next Sections, we will use a representative subset of the ratio of mobile to fixed nodes, namely R_n= 1, 2, 3 and 4.

The X-MAC$_{mnbcast}$ scheme hardly scales to a large number of mobile nodes. Mobile sensors sending broadcast data at the layer 2 have to send the whole strobed preamble. Indeed, the preamble frames sent prior to the data being also transmitted in a broadcast fashion, it cannot be acknowledged by neighbor nodes. This reduces the medium availability which can result in congestions. Furthermore, data packets from the mobile nodes are not acknowledged neither. The mobile sensor has no way to know whether its packet was received by at least one neighbor, hence cannot trigger any retransmissions. Finally, duplications can occur as data from the mobile node might be received and forwarded by multiple fixed sensors. In the experiment with 16 mobile nodes operating X-MAC$_{mnbcast}$ (R_n= 1), as much as 48% of the received packets are actually duplicated ones. This reduces all the more the channel availability. A thorough analysis of the losses at the MAC layer will confirm these facts.

5.2.3 *Losses at the MAC layer*

The average data packet losses at the MAC layer as well as the causes on the receiver side are depicted in Figure 40.

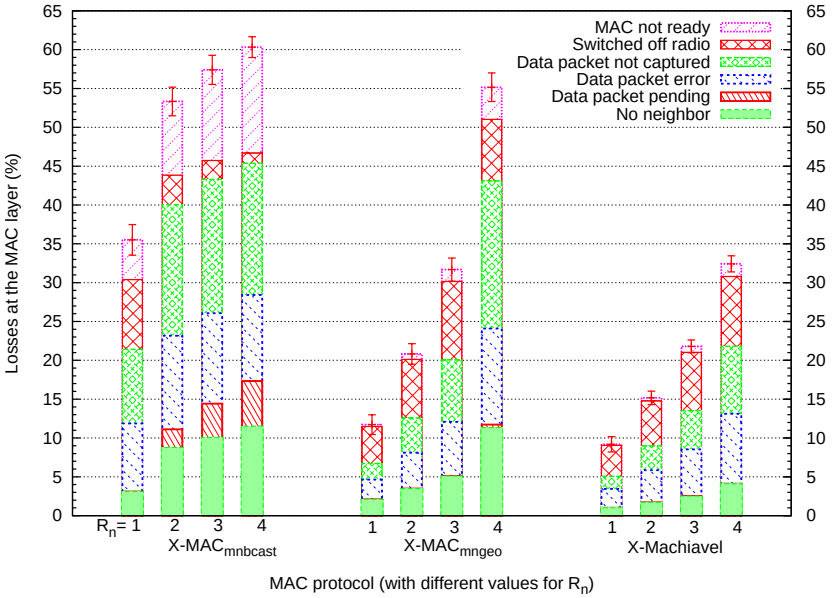


Figure 40: Average losses at the MAC layer according to R_n.

MAC not ready means that the peer was not in a proper state to correctly receive the data. This cause mainly occurs in the X-MAC$_{mnbcast}$ case: all neighbors of the mobile nodes are potential receivers of the data but may not be ready to receive it.

Together with the *switched off radio*, it illustrates a synchronization issue between the sender and the receiver. Most likely, the receiver did not capture the preamble correctly (due to a collision or a packet error) and thus switched off its radio prematurely. A *data packet not captured* means that the radio has captured another packet (preamble or data) instead, because the reception sensibility was better for that packet. A *data packet error* occurs when the packet was captured correctly, but dropped upon reception because it contained errors caused by a collision. These four causes initially happen because the receiver node hears multiple transmitters at the same time although one of the role of the preamble is to reserve the medium. Note that X-MAC and subsequently X-Machiavel do not implement any mechanism (such as RTS/CTS) to reduce the hidden node problem. The main reason resides in the size of data packets in WSN, which is usually similar to the one of a RTS packet. It is thus preferable to send directly the data rather than using a RTS/CTS handshake.

We can however note that X-Machiavel manages to reduce the proportion of such losses, especially when $R_n = 4$. When a mobile node operating X-Machiavel takes possession of an occupied medium (as described in Section 5.1.3), the sender of the preamble and the receiver of the data from the mobile sensor are the same node: the forwarder. In this configuration, no hidden nodes can interfere with the reception of the data from the mobile node. Indeed, the preamble sent by the forwarder prevents its neighbors from emitting on the medium. It is thus likely that no collision will occur at the forwarder with the data from the mobile node.

No neighbor means that the packet could not reach its destination because the receiver was not in the vicinity of the sender anymore. In the X-MAC$_{\texttt{mnbcast}}$ and X-Machiavel cases, this can happen when fixed nodes try to reach a mobile node: the next hop chosen by the geographic routing algorithm may not be valid anymore when the node actually sends the packet on the medium. This is especially true if packets are queued for some time at the MAC layer. Such issue can also happen on mobile nodes in the X-MAC$_{\texttt{mngeo}}$ case. As a result, the whole strobed preamble is sent without being acknowledged (which reduces the medium availability) and the ensuing data is lost. Furthermore, no layer-2 acknowledgments are ever received for the data packet, which triggers a reemission until the retransmission threshold is reached (it is set to 3 in our experiment). As the reemitted data may be lost for the same reasons, the proportion of *No neighbor* losses are needlessly increased. However, X-Machiavel manages to mitigate such losses on the mobile nodes because the next hop is set at the layer 2 upon effective data transmission (as detailed

in Section 5.1.2 and 5.1.3). Still, a similar mechanism on the fixed sensors would allow to further reduce these losses.

Data packet pending indicates that some packets were not delivered because they were still queued on a node at the MAC layer at the end of the simulation (i.e. even after the 300 s cool-down period). This issue mainly occurs for the X-MAC$_{mnbcast}$ case, and explains the significance of the losses due to the *No neighbor* reason: packets experience long delays in queues before being sent on the medium.

Although X-MAC$_{mngeo}$ and X-MAC$_{mnbcast}$ face roughly the same order of losses at the MAC layer with $R_n = 4$, X-MAC$_{mngeo}$ can attenuate the losses at the application layer thanks to the layer 2 retransmissions on the mobile nodes. By decreasing MAC losses in dense topologies (by almost a factor of 2 when $R_n = 4$), X-Machiavel can improve the data delivery ratio as seen in the previous Section.

5.2.4 *Medium access delay*

The average delay per packet to access the medium is depicted in Figure 41. In addition, separate values for the fixed and mobile sensors are also detailed. This one-hop delay includes the initial backoff, the channel Sampling Period, potential congestion backoffs, and the Preamble Length.

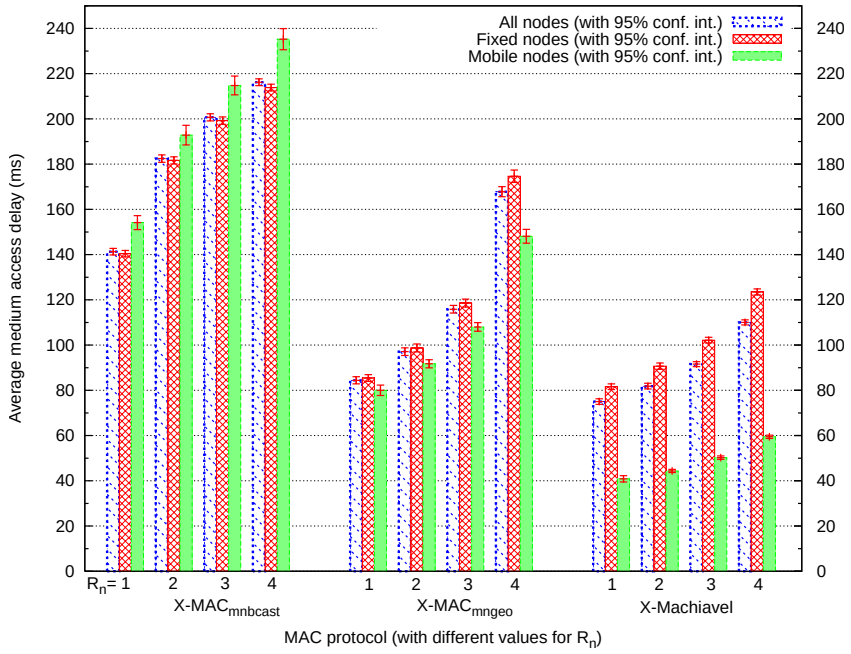


Figure 41: Average medium access delay according to R_n.

When $R_n = 4$, a mobile node operating X-MAC$_{mnbcast}$ can take as much as 235 ms in average before being able to send its data packet. Due to the layer-2 broadcast, it has to send the whole

preamble (100 ms) which accounts for a large fraction of the delay. Most of the other part is caused by multiple congestion backoffs. Since broadcast data packets are received and forwarded by multiple peer nodes, the competition increases on the medium. As a consequence, a node is less likely to send its data once its initial backoff has expired. Even though fixed sensors may have their strobed preamble acknowledged earlier, their average medium access delay is fairly similar to the one experienced by the mobile sensors. This can be explained by the high contention on the channel (due to the long broadcast preamble sent by mobile nodes), which greatly increases the length of the congestion backoff on the fixed nodes.

Up to $R_n = 3$, both X-MAC$_{mngeo}$ and X-Machiavel exhibit a medium access delay decreased by a factor 2 compared to X-MAC$_{mnbcast}$. X-Machiavel achieves from 10 to 20 ms better than X-MAC$_{mngeo}$ on average. For $R_n = 4$, we can note a clear advantage for X-Machiavel. Especially, we can point out that the medium access delay on the mobile node is divided by a factor 2 when operating X-Machiavel. By authorizing the mobile sensors to take possession of the medium in contended areas, and allowing the first available forwarder to claim the data of the mobile node, X-Machiavel manages to alleviate the time a mobile sensor needs to effectively sends its data on the medium. Reducing this delay increases the medium availability, hence decreases the congestion backoff on the fixed nodes. In addition to the benefits brought by the multi-hop operations described in Section 5.1.4, this explains why the average 1-hop delay on the fixed sensor is also improved when operating X-Machiavel.

We can however note that fairness in terms of medium access delay is not achieved between fixed and mobile sensors: for each value of R_n, fixed nodes experience a delay that is roughly twice the one of a mobile node operating X-Machiavel. Yet, we can point out that they still achieve better than when X-MAC$_{mngeo}$ or X-MAC$_{mnbcast}$ is operated in the network.

5.2.5 *Multi-hop performances*

In order to better understand what is the impact of X-Machiavel on multi-hop communications, we first depict in Figure 42 the distribution of the layer-2 communication patterns during the experiment.

We can note that X-MAC$_{mnbcast}$ exhibits a large proportion of mobile-to-mobile communications, especially for high values of R_n. Actually, the preamble and data from the mobile sensor being sent in a broadcast fashion, all the nodes in range will receive the data packet, even if they are not the final destination. With high values of R_n, the neighborhood of the mobile sensor will

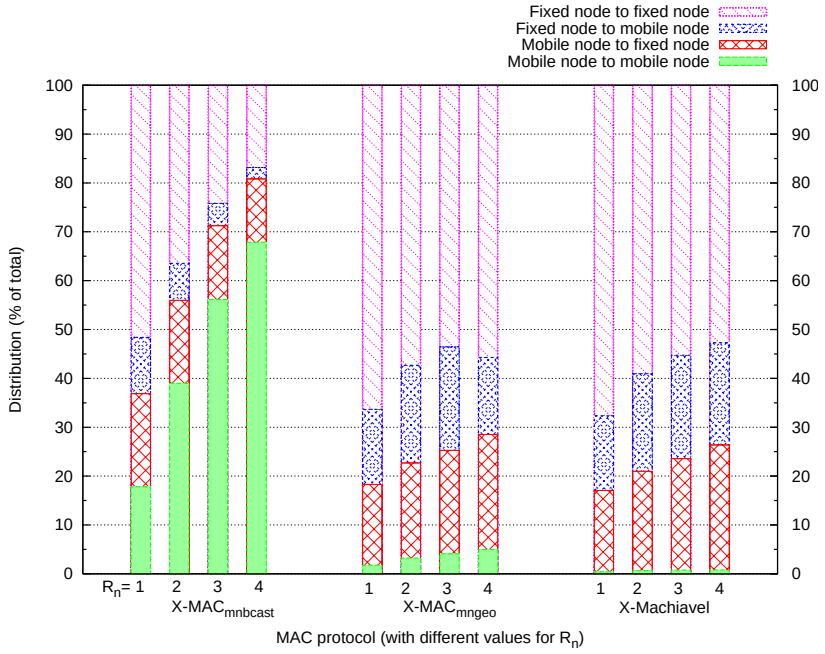


Figure 42: Distribution of the communication patterns.

be mainly composed of mobile nodes, hence the large proportion taken by mobile-to-mobile communications. Because other patterns (mobile-to-fixed, fixed-to-mobile and fixed-to-fixed) use unicast communications, their proportions decrease when R_n increases.

With X-MAC$_{mngeo}$ and X-Machiavel, we can however state that all the mobile-to-mobile communications actually correspond to a direct transmission between a mobile node and its final destination. X-MAC$_{mngeo}$ shows a higher percentage of such communication pattern. Indeed, a fixed node operating X-Machiavel can claim the data of a mobile node, which may prevent direct communication from occurring if the remote peer was actually in the vicinity of the mobile sensor.

As raised in Section 5.1.2, such behavior can have an impact on the number of hops between a mobile node and its peer. We have thus computed in Figure 43 a distribution of the number of hops that data packets have to go through to reach their final destination.

Compared to X-MAC$_{mngeo}$, X-Machiavel has a lower proportion of 1-hop distance communications. This confirms that X-Machiavel reduces direct communications between mobile nodes. In addition, when a data packet has to travel through multiple hops to reach its destination, mobile nodes operating X-Machiavel may not always choose the best next hop compared to the ones operating X-MAC$_{mngeo}$. While the random geographic routing guarantees the mobile node that its next hop is nearer from the destination than itself, a mobile node operating

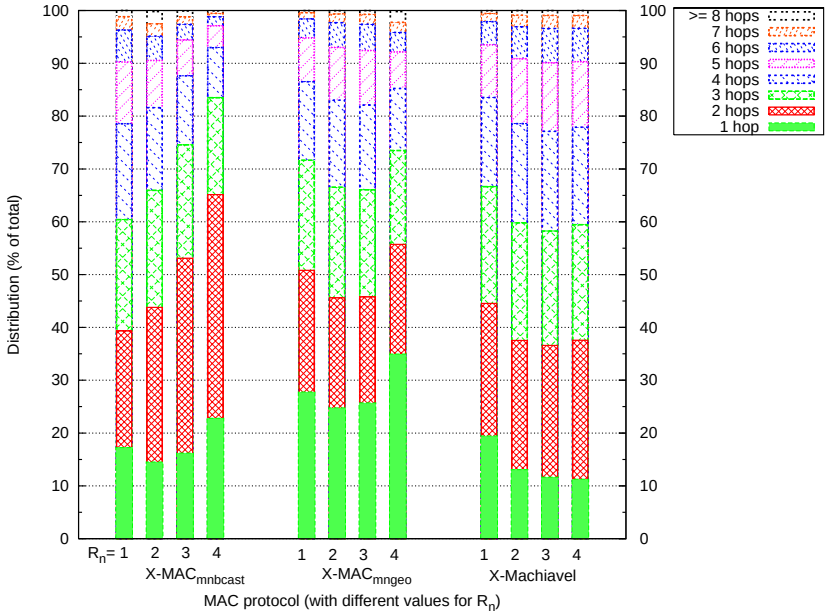


Figure 43: Average distance to the destination in number of hops.

X-Machiavel does not know whether its forwarder is closer to the destination. For these reasons, data packets can experience longer distances when X-Machiavel is operated on the nodes.

In order to analyze to what extent this penalizes X-Machiavel in terms of end-to-end delay, we detail the average latency between nodes according to the number of hops in Figure 44. It only depicts $R_n = 1$ for X-MAC$_{mnbcast}$, because higher ratios experience very long delays, even at the first hop. For example, when $R_n = 2$, the delay at one hop is 74.5 s ($\pm$ 2.4), and reaches

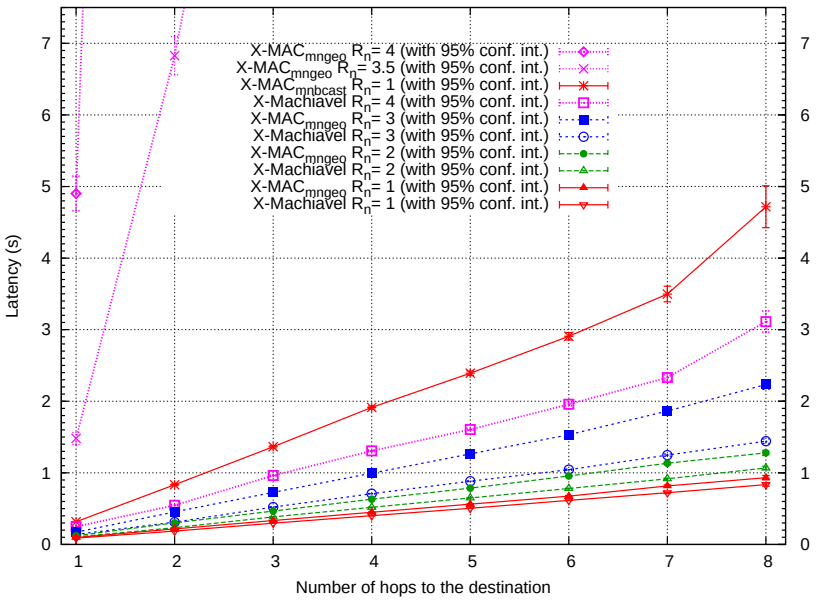


Figure 44: End-to-end delay between peer nodes.

261.6 s (± 3.8) for 3 hops. The large difference observed with the previously computed medium access delay (which was not exceeding hundreds of ms) comes from the time spent in the MAC queue, and suggests serious congestions in the network.

X-MAC$_{mngeo}$ and X-Machiavel exhibit similar results when R_n is equal to 1 or 2, with slightly better performances for X-Machiavel over long distances. When R_n= 3, X-Machiavel decreases the delay by around 30% at every hop compared to X-MAC$_{mngeo}$. For R_n= 4, X-Machiavel outperforms X-MAC$_{mngeo}$ which takes as much as 4.9 s (± 0.2) on the first hop, and 84.7 s (± 1.5) at 3 hops. As a matter of comparison, we have also displayed the results for X-MAC$_{mngeo}$ when R_n= 3.5. It still achieves worse that X-Machiavel when R_n= 4. By reducing the MAC layer losses and the medium access control delay, as well as defining specific multi-hop operations for the fixed nodes (as detailed in Section 5.1.4), X-Machiavel manages to improve the end-to-end delay in a significant manner, especially in highly mobile environments.

We can point out that even though X-Machiavel increases the number of hops toward a destination, the overall delay to reach the remote peer is tempered by the gain across multiple hops. For example, with R_n= 3, it almost takes as much time for a packet to travel through 6 hops with X-Machiavel (1.04 s) than it takes for 4 hops with X-MAC$_{mngeo}$ (0.99 s).

5.2.6 *Energy consumption*

Energy consumption being one of the crucial point in WSNs, we also evaluate the overall energy depletion in the network at the end of the simulation. It is depicted in Figure 45.

As expected, X-MAC$_{mnbcast}$ is the less energy efficient protocol. As pointed in Table 10, radio reception constitutes the main reason of the total energy consumed by the fixed nodes in the network (70.7% when R_n= 4). Indeed, the long broadcast preamble sent by mobile sensors prevents the neighbor nodes from switching off their radio prematurely: they have to listen to the whole preamble to receive the trailing data.

X-Machiavel and X-MAC$_{mngeo}$ present similar results for R_n values up to 2. However, X-Machiavel consumes 11.7% less energy when R_n= 3. This reduction reaches 33% when R_n= 4. The energy consumed due to slightly longer routing paths with X-Machiavel is balanced with the savings performed on the mobile node which sends reduced preambles (when a forwarder claims its data) or no preamble at all (in the case the mobile sensor takes possession of the medium as described in Section 5.1.3). Radio transmissions constitutes only 3.75% of the energy consumed by mobile nodes operating X-Machiavel, as shown in Table 10. The reduction of

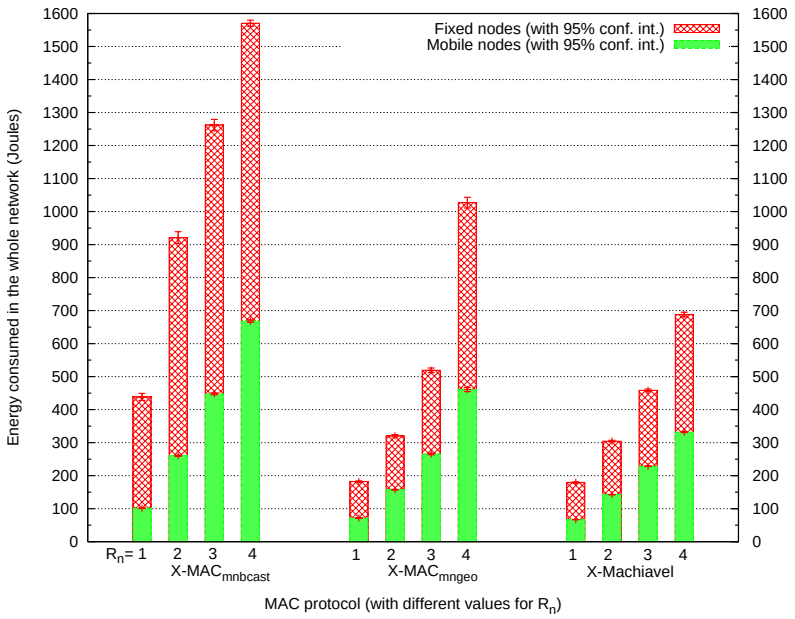


Figure 45: Energy consumed in the network at the end of the simulation.

the layer-2 losses as detailed in a previous Section also decreases the packet retransmissions which are energy-consuming.

5.3 CONCLUSION

So far, only a few protocols have considered the specificities of mobile environments, and no one really satisfies scenarios where the mobile nodes numerically exceed the fixed ones. We have particularly focused on such topologies with the contribution presented in this Chapter. The X-Machiavel protocol relies on an infrastructure of fixed sensors which takes care of routing the data between peers. On an idle channel, fixed nodes can claim the data of a mobile sensor and route it toward its destination.

MAC protocol	*Type of nodes*	*Total energy consumed in the network (Joules)*	*Main reasons (% of total consumed)*			
			Tx	*Rx*	*Idle*	*Radio init.*
X-MAC$_{mnbcast}$	fixed	902.01 (± 9.24)	19.14	70.7	9.48	0.68
$R_n = 4$	mobile	668.42 (± 3.88)	4.32	56.6	18.6	20.48
X-MAC$_{mngeo}$	fixed	565.91 (± 16.22)	20.0	59.2	17.33	3.47
$R_n = 4$	mobile	461.16 (± 6.43)	8.21	40.14	20.55	31.1
X-Machiavel	fixed	355.92 (± 7.03)	25.24	43.85	23.41	7.51
$R_n = 4$	mobile	331.99 (± 1.48)	3.75	29.86	22.05	44.34

Table 10: Average energy spent in the network at the end of the simulation when $R_n = 4$ (with 95% confidence interval).

On a congested medium, mobile nodes can take possession of the channel initially reserved by a fixed sensor in order to fit in the communication schedule. Lastly, by setting the layer-2 next hop just before the effective data transmission without the need for a routing protocol, mobile nodes are more likely to reach their peers.

Through an evaluation by simulation, we have demonstrated the benefits of our solution compared to X-MAC on which our proposal relies. X-Machiavel outperforms X-MAC when the latter is used in association with a simple layer-2 broadcast scheme on the mobile nodes. When X-MAC is combined with a geographic routing protocol on the mobile sensors, X-Machiavel exhibits substantial improvements when the number of mobile nodes increases in the network. Especially, the fact that the routing information may not be valid upon effective data transmission with X-MAC gives a true advantage to X-Machiavel in congested networks. By mitigating the losses at the MAC layer, X-Machiavel also improves the end-to-end delivery ratio. Even though X-Machiavel may increase the number of hops between nodes, we have shown that it is not performed at the cost of the energy-efficiency nor the end-to-end delay, especially in highly mobile scenarios. Network-wide, the overhead for the fixed sensors is balanced by the gain achieved on the mobile ones.

We have however spotted several possible optimizations. For example, communications from the fixed nodes to the mobile sensors could be further improved in order to avoid useless retransmissions when the mobile node is not in the vicinity of the sender anymore. It could also be interesting to evaluate our solution with other data collection schemes. We expect convergecast traffic to particularly benefit from the multi-hop operations of X-Machiavel. Our contribution relies on an infrastructure of fixed sensors, it could be interesting to study to what extend we could apply our scheme to networks composed of only mobile nodes (e.g. by classifying mobility at different degrees). Note that the principles behind X-Machiavel can be used with other preamble sampling protocols in order to improve their efficiency in mobile environments.

Part III

FROM VERSATILITY TO AUTO-ADAPTATION

6 PERFORMANCES OF VERSATILE PROTOCOLS

Contents

Recent deployments of wireless sensor networks have targeted challenging monitoring and surveillance applications. The medium access control being the main source of energy wastage, energy-efficiency has always been kept in mind while designing the communication stack embedded in spread sensors. Especially, versatile protocols have emerged to offer a suitable solution over multiple deployment characteristics.

In this Chapter, we analyze to what extent versatility applies to a dynamic scenario in which antagonist traffic patterns exist in the network. We provide a performance evaluation of two well-reputed versatile protocols (B-MAC [77] and X-MAC [17]) under the conditions of the scenario described in Chapter 2 (Section 2.1), in which some parts of the network alternate sporadic and burst transmissions.

6.1 SIMULATION ENVIRONMENT

In order to shed some light on the issues that we outlined in Chapter 2, we have used the WSNet software. An example of the topology used in our simulations is depicted on Figure 46. The simulation scenario consists in a large grid network of hundred (10x10) wireless sensor nodes. Each node is located two meters from its neighbors. The sink is located at the bottom left corner of the grid. As we explained in Section 2.1, all the sensors in the network first operate in a event-driven manner, and thus do not send any data as long as they do not catch an event. An event is stimulated on a sensor by feeding him with a measured value greater than a certain threshold. We have distributed the starting time of 180 events over the simulation time (3 hours) according to a Poisson process:

$$\lambda poisson := \frac{180}{10800}$$

which corresponds to 180 events distributed on 10800 time-slots (3 hours time-frame divided into 1-second slots). The location of the event is uniformly distributed in the sensor grid. Upon an event, a node starts sending data packets in a burst fashion toward the sink. It sends one packet every second during the length of the event, which lasts 10 s.

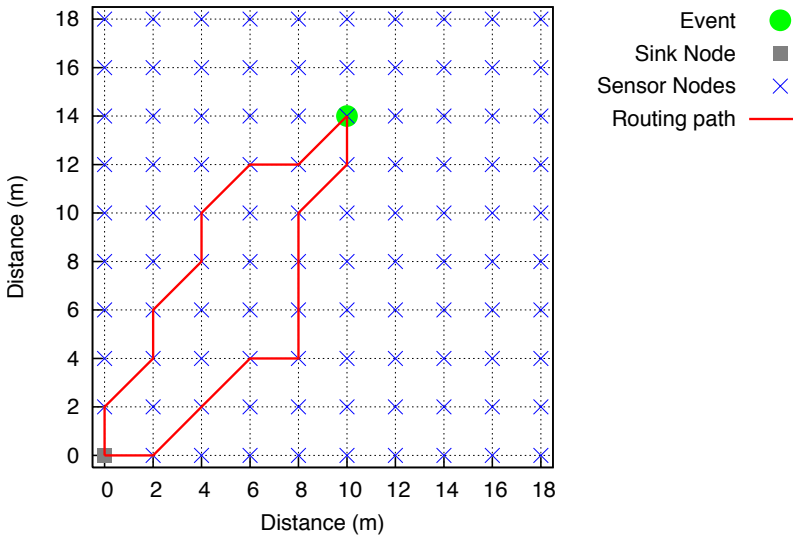


Figure 46: The topology used in our simulations. As an example, an event is triggered at node [10;14]. Two packets are routed through different paths toward the sink using a random geographic routing.

On the protocol side, the details of the models used in our simulations are exposed in Table 11. With the random geographic routing, a node forwarding a packet selects its next hop randomly among the set of nodes that are strictly nearer from the destination than itself. In order to evaluate the performances of the B-MAC and X-MAC protocols, we vary their LPL values (i.e. their Preamble Length PL and Sampling Period SP) from 100 ms to 500 ms. The same pre-configured value is used on all the nodes throughout the simulation. The energy model is configured from the consumption of a Chipcon CC1100 chipset [94]. As the radio sleep current is 400 nA (which is about 40 000 times less than the reception current), we consider it as negligible in our simulations. The energy consumption results presented in the following Sections make the assumption that a 3 V battery is used (for example two 1.5 V AA cells, as commonly used on TelosB or Mica2 motes [24]). Other models (modulation, propagation, radio and antenna) have been detailed in Chapter 3 (Section 3.1).

Each possible configuration of MAC layers and LPL values has been simulated 20 times with a different distribution of the events. The results outlined in the next Section are an average of the over-

Simulation parameter	*Value*
Topology	Regular square grid (20x20 m) of 100 (10x10) fixed sensors
Data sending period	Upon an event: every second during 10 s
Data packet/payload size	25 Bytes / 4 Bytes
Routing model	Random geographic routing
MAC model	B-MAC or X-MAC with a LPL of 100, 250 or 500 ms
Radio model	Half-duplex, bandwidth: 15 kB/s
Antenna model	Omnidirectional
Radio propagation model	Friis, 868 Mhz, pathloss: 2 (range: 3 m)
Modulation model	BPSK
Energy model With a 3V battery	Idle: 1.6 mA, Rx: 15 mA, Tx: 16.9 mA, Radio init: 8.2 mA Idle: 4.8 mW, Rx: 45 mW, Tx: 50.7 mW, Radio init: 24.6 mW
Number of events	180 events
Simulation setup	3 hours, simulated 20 times

Table 11: Simulation parameters used in our experiments.

all data collected on the set of simulations. The 95% confidence interval exposed in our results confirms the reliability of our measures.

6.2 RESULTS

The results exposed hereinafter exhibit the impact on energy consumption, end-to-end and one-hop delays as well as packet losses.

6.2.1 *Energy consumption*

Figures 47 and 48 depict a map of the energy consumption of our sensor grid at the end of the simulation for B-MAC and X-MAC respectively. Results have been computed for the three LPL values (100, 250 and 500 ms). We can observe that when a short preamble is used (100 ms), the energy consumption is much more uniform in the network than with a longer preamble (250 ms or 500 ms), for both B-MAC and X-MAC. The length of the preamble has a direct impact on several energy-consuming factors.

First, each data packet sent or forwarded by a node is preceded by a preamble. The time spent to send the actual data is pretty small compared to the size of a preamble. For example, the size

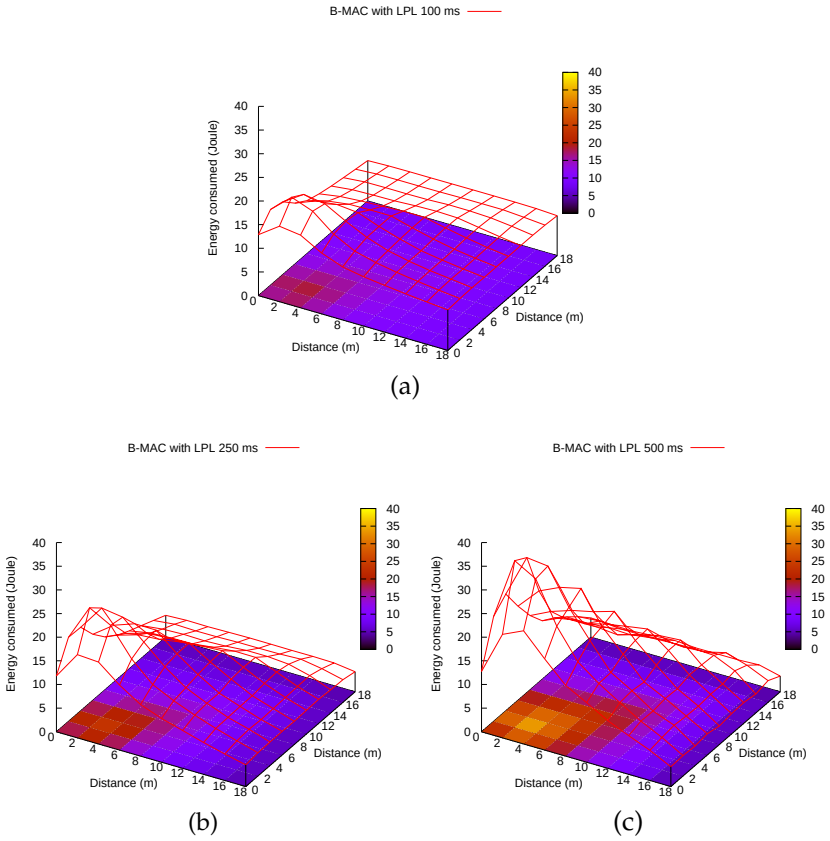


Figure 47: Map of the energy consumption in the sensor grid at the end of the simulation for B-MAC for three LPL values: (a) 100 ms, (b) 250 ms, (c) 500 ms.

of a packet in our experiment is 25 bytes (4 bytes of data payload, and 21 bytes of protocol headers), which takes around 1.7 ms to be sent over the medium. The preamble is thus the main factor of energy consumption when a packet is transmitted or received. Longer preambles can thus increase transmission and reception costs in a non-negligible manner, as displayed in Table 12. An exception for X-MAC is that the energy spent in reception remains fairly below other sources of energy consumption (namely transmission, idle listening and radio initialization) whatever the LPL value employed. Indeed, nodes operating the X-MAC protocol can send an early acknowledgment to the transmitter as soon as they hear a piece of the preamble, and even switch off their radio if the preamble information does not identify them as the target of the ensuing data. This greatly reduces the overall time spent in reception in the network.

Second, the Sampling Period on each node is the same as the Preamble Length (SP = PL). The smaller the preamble, the more often nodes sample the medium. This operation mandates to switch on, calibrate and initialize the radio, which consumes a

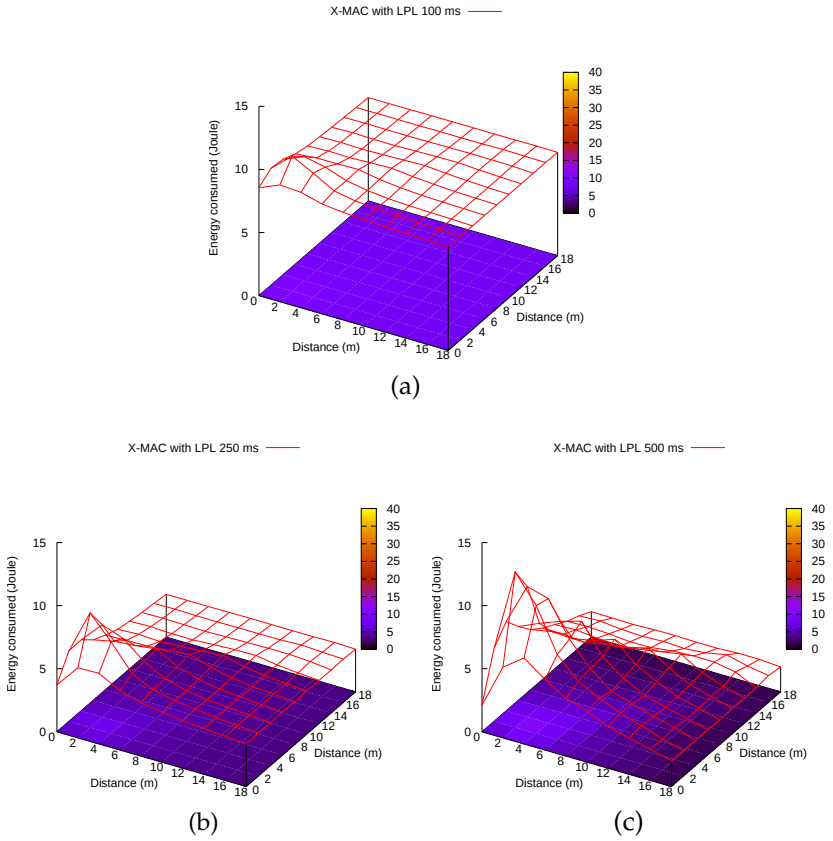


Figure 48: Map of the energy consumption in the sensor grid at the end of the simulation for X-MAC for three LPL values: (a) 100 ms, (b) 250 ms, (c) 500 ms.

non-negligible amount of energy (24.6 mW as stated in Table 11). Table 12 points out that this is actually the major reason of energy consumption in our simulations when a small SP is used. It can be as high as 61% (respectively 77%) of the overall energy consumption in the network when B-MAC (respectively X-MAC) is used with a PL of 100 ms. Therefore, when the PL is short, the energy cost is increased on the whole network due to more periodic channel samplings. This is corroborated by the Figures 47 and 48 for a LPL value of 100 ms. The overhead due to the transmission and reception of preambles is however minimized thanks to its short size. The energy saved by sending smaller preambles is thus lost by more regular samplings. On the opposite, when a longer preamble is used (250 ms or 500 ms), the energy cost is much higher in places where nodes forward more packets (typically around the sink) but greatly decreases in other places as channel sampling is performed less often.

The total energy consumed in the network (Table 12) reveals that B-MAC with a LPL of 250 ms has consumed less energy than with other configurations. With X-MAC, using a LPL of 500 ms is

MAC protocol	*Total energy consumed in the network (Joules)*	*Main reasons (% of total consumed)*			
		Tx	*Rx*	*Idle*	*Radio init.*
B-MAC (LPL 100 ms)	1094.67 (± 2.69)	7.48	18.4	12.66	61.46
B-MAC (LPL 250 ms)	972.22 (± 6.29)	22.91	42.07	7.54	27.49
B-MAC (LPL 500 ms)	1231.31 (± 10.78)	35.07	48.62	5.54	10.77
X-MAC (LPL 100 ms)	847.3 (± 0.44)	2.31	0.87	19.48	77.34
X-MAC (LPL 250 ms)	406.45 (± 1.45)	11.92	2.97	19.53	65.57
X-MAC (LPL 500 ms)	380.48 (± 3.19)	30.62	15.22	19.11	35.04

Table 12: Average energy spent in the network for X-MAC and B-MAC at the end of the simulation (with 95% confidence interval).

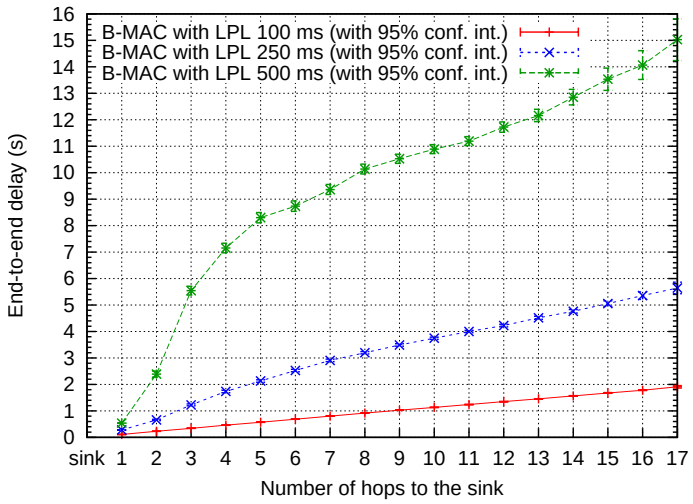
more efficient network-wide. Let us assume that the data traffic would always follow the same pattern than during the simulated 3 hours, and that each node uses two 1.5 V AA 2500 mAh cells (as commonly used on TelosB or Mica2 motes). In that case, the experiment with B-MAC and a 250 ms LPL could last around 350 days before all nodes would die, and 136 days before the first node dies. With X-MAC and a 500 ms LPL, the overall energy available in the network would be depleted within 900 days, while the most consuming node would die after 280 days.

As a side note, these experiments confirm that X-MAC is more energy-efficient than B-MAC thanks to its strobed preamble and early acknowledgment mechanism. Actually, this is only the case when unicast packets are sent or forwarded by sensor nodes (such as in our scenario). In case of broadcast packets, the early acknowledgment could not be used, hence energy consumption (as well as the other results presented later) would be roughly the same for both protocols.

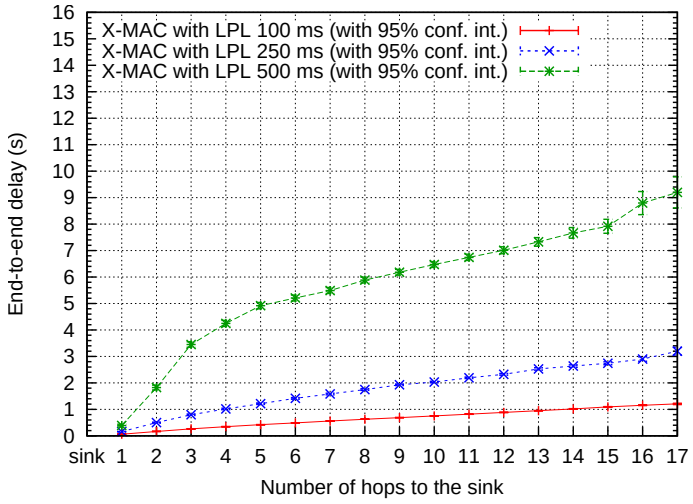
6.2.2 *End-to-end and one-hop delays*

The average end-to-end delay according to the number of hops to the sink is depicted in Figure 49. For both B-MAC and X-MAC, higher LPL values imply an increase in the node-to-sink delay. Indeed, this delay is increased at least by the length of the preamble each time the data packet is forwarded by a node on the routing path. While the results are nearly linear for LPL values of 100 and 250 ms, we can notice a quick increase close to the sink when a 500 ms preamble is used. For example, it takes 5.5 s to go through the last 3 hops in a network operating B-MAC (3.4 s for X-MAC). This suggests congestions around the sink when high LPL values are used.

In order to further investigate this issue, we have computed the average delay per packet to access the medium for B-MAC and



(a)



(b)

Figure 49: End-to-end delay according to the number of hops to the sink for (a) B-MAC and (b) X-MAC.

X-MAC in the different LPL conditions. The results are detailed in Table 13. This one-hop delay includes the initial backoff, the time to sample the channel, potential congestion backoffs and the size of the preamble. A node operating B-MAC (respectively X-MAC) with a 500 ms preamble can take as much as 700 ms (respectively 433 ms) in average before being able to send or forward a packet. This reveals that B-MAC can hardly handle more than one packet per second (two packets per second for X-MAC) without provoking congestions. Sensor nodes located around the sink suffer from the *funneling effect*: they have to forward packets coming from all the network. As a consequence in our scenario, as soon as two events occur in the same 10-second timeframe, sensor nodes around the sink may have to forward

MAC protocol	*Average one-hop delay (ms)*		
	LPL 100 ms	*LPL 250 ms*	*LPL 500 ms*
B-MAC	106.98 (± 0.16)	288.15 (± 1.12)	699.67 (± 3.81)
X-MAC	72.21 (± 0.55)	177.63 (± 1.80)	433.31 (± 2.00)

Table 13: Average one-hop delay (with 95% confidence interval) for B-MAC and X-MAC.

packets at a rate above the congestion limit. This results in the increase of the one-hop delay near the sink.

Long preambles reduce the channel availability, hence causing congestions. This can have an impact on the packet losses that we propose to study in the next Section.

6.2.3 *Packet losses*

We have first computed the losses at the sink. In each experiment, a total of 1800 packets have been sent at the application layer by the nodes (180 events are generated during the simulation, each inducing 10 data packets). The number of packets received at the sink as well as the resulting percentage of losses are detailed in Table 14. We can confirm that higher LPL values are the cause of more packet losses.

In order to better understand why packets are dropped, we expose in Table 15 the reasons of the losses at the radio level. The three main ones are detailed below. A *switched off radio* while receiving a data packet illustrates a synchronization issue between the sender and the receiver. Most likely, the receiver did not capture the preamble correctly (due to a collision or a packet error) and thus switched off its radio prematurely. Another cause for nodes operating X-MAC is that the receiver heard a preamble that was not targeting it (and thus switched off its radio) although another peer node was sending a preamble destined to it. A *data*

MAC protocol	*Packet sent by applications*	*Packet received at the sink*	*% of losses at the sink*
B-MAC (LPL 100 ms)	1800	1773.05 (± 4.34)	1.49
B-MAC (LPL 250 ms)	1800	1518.4 (± 12.42)	15.64
B-MAC (LPL 500 ms)	1800	1289.65 (± 13.48)	28.35
X-MAC (LPL 100 ms)	1800	1795.65 (± 1.71)	0.24
X-MAC (LPL 250 ms)	1800	1691.45 (± 11.56)	6.03
X-MAC (LPL 500 ms)	1800	1333.9 (± 12.14)	25.89

Table 14: Average packet losses at the sink (with 95% confidence interval) for B-MAC and X-MAC.

MAC protocol	Packets forwarded by the MAC	Packets received at the MAC	% of losses	Main reasons (%)		
				Radio off	Packet not captured	Packet error
B-MAC (LPL 100 ms)	15611.85	15229.1	2.4	70.2	9.3	20.0
B-MAC (LPL 250 ms)	17322.05	13536.55	21.8	80.2	6.6	12.7
B-MAC (LPL 500 ms)	16901.75	11886.65	29.6	75.8	8.7	14.8
X-MAC (LPL 100 ms)	15527.55	15393.3	0.8	61.4	7.5	30.0
X-MAC (LPL 250 ms)	16614.1	14943.3	10.0	66.2	6.8	24.5
X-MAC (LPL 500 ms)	19214.4	13379.1	30.3	63.1	7.6	25.5

Table 15: Average losses at the MAC layer and main reasons for B-MAC and X-MAC. Main reasons are given as a percentage of the total of packet losses.

packet not captured means that the radio has captured another packet (preamble or data) instead, certainly because the reception sensibility was better for that packet. A *packet error* happens for a data packet when it was captured correctly, but dropped upon reception because it contained errors caused by a collision.

All of these three causes initially happen because the receiver node hears multiple transmitters at the same time. Although one of the role of the preamble is to reserve the medium, we do not use any mechanism (such as RTS/CTS) to reduce the hidden node problem. This issue is emphasized when high LPL values are used, because congestions increase the channel utilization. We can affirm that it has a direct impact on the energy consumption in the network. A packet loss at the MAC layer triggers a retransmission as long as the threshold has not been reached. This limit has been fixed to 3 times in our simulations. It contributes to waste energy at both the sender and the receiver, especially when long preambles are used, as outlined in Section 6.2.1.

6.3 CONCLUSION

From the results exposed in this Chapter, we can notice that there are no perfect configurations regarding the LPL values when we confront preamble sampling protocols to a scenario in which burst transmissions randomly occur in space and time. Advantages and drawbacks of different LPL values are summarized in Table 16. When small preambles are used, a lot of energy is wasted on all the nodes in the network due to more regular channel samplings. It however allows forwarding nodes to reduce the time they spend in transmission and reception, hence saves energy on the routing path. This is the exact opposite when long preambles are configured on the sensor nodes. The node-to-sink delay increases with the LPL values, a better reactivity can thus

LPL *value*	*Advantages*	*Drawbacks*
Low	Better reactivity, energy savings on routing paths	Energy wastage on idle places
Medium	Trade-off between energy and delays	Not optimized in idle places nor routing paths
High	Energy savings on idle places	Long delays, energy wastage on routing paths, congestions

Table 16: Advantages and drawbacks of various LPL values.

be achieved with smaller preambles. Long preambles make the medium less available, which can provoke congestions around the sink and emphasize the funneling effect. This has a direct impact on the packet losses which increase with the size of the preamble.

The LPL value of 250 ms could be a trade-off between the energy consumption, end-to-end delays and packet losses. Still, in that configuration the link-layer costs are high compared to what could be achieved if we could mix the advantages of both the low and high LPL values. As already discussed in Chapter 2 (Section 2.3.1), using different LPL values on distinct node can prevent them from communicating. We thus propose in the next Chapter an algorithm that allows to adapt dynamically the PL and SP on every node while ensuring connectivity among them.

7

AUTO-ADAPTIVE MAC FOR ENERGY-EFFICIENT BURST TRANSMISSIONS

Contents

The results obtained in Chapter 6 convinced us to propose more than versatility and pre-configured solutions, that is auto-adaptation. We insist on how different versatility and auto-adaptation are. Interfaces available in MAC layers such as B-MAC or X-MAC enable versatility. We aim at introducing auto-adaptation, that is using this versatility feature to automatically tune the LPL mechanism and optimize energy savings.

In this Chapter, we introduce our contribution [52], an auto-adaptive algorithm that allows to adjust the previously mentioned protocols while the network is operating. Our solution is based on a link-by-link negotiation in order to establish energy-efficient communications over certain paths toward the sink station. This mechanism, auto-adapting to the current network traffic, can be used in combination with various other preamble-based MAC layers and works for every traffic pattern, being especially energy-efficient during burst transmissions. After detailing our contribution, we will analyze to what extent it outperforms the results obtained in Chapter 6.

7.1 ALGORITHM OVERVIEW

7.1.1 *Main characteristics*

With LPL implemented, a long preamble leads to much energy spent by senders while a short one imposes all sensors to reduce their sleeping periods, even if no data is to be sent, thus increasing global power consumption. In order to favor energy-efficiency during burst transmissions, our goal is to enhance data communication paths along which each relaying node adopts a short preamble thus reducing energy spent during transmissions. Nodes are expected to switch between a long and a short PL and SP according to their participation in multi-hop communications, either as the main source of data or as simple relaying nodes. Nodes not involved in any communication paths keep sleeping for long periods of time, thus saving energy.

The paths leading from the data sources to the sink station will then be composed of Energy-Efficient (EE) links. A sending node is either a data source (located nearby an event for instance) or a relay node participating in multi-hop communications. The two different durations for PL and SP are assumed to be fixed prior to the deployment: they are noted as T_{max} for the long one, and T_{min} for the short one. As opposed to B-MAC and X-MAC, one characteristic of our solution is that PL and SP can take different values on the same node. PL is managed when the node is in sending mode (Section 7.1.2) while SP is configured in reception mode (Section 7.1.3). The key contribution of our solution is that two nodes u and v configured with different PL and SP will not be desynchronized (as opposed to what we described in Chapter 2, Figure 17).

7.1.2 *Operations on the sender side*

When a sender has data to transmit, it sends the first packet with a long preamble (PL = T_{max}). In order to create an EE link, both ends must agree on a short preamble (PL = T_{min}). This negotiation is kept simple and is initiated by the first data transmission. Our algorithm assumes that any sender aims at reducing the size of its preamble in order to save energy during further transmissions. By this way, upon reception, each receiver knows that the sender wants to reach a short preamble.

In order to enable this short preamble, the receiver will acknowledge the data reception. This acknowledgment is considered as an agreement by the sender. Further communications will therefore only require an energy-efficient short preamble. This agreement should not stand for too long a time, we would otherwise fall back in the case where sensors uselessly sleep for

short periods of time while no transmission is ongoing. To avoid this problem, agreements have a validity period. Every sender u keeps an up-to-date list of the receivers that have agreed on a short preamble. This list is later referred to as $Receivers_u$. Every entry of this list is attributed a timeout. Once this timeout ends, the sender knows that the corresponding receiver may no longer be using a short SP and thus needs to initiate the agreement once again. For any transmission to node v, a node u checks $Receivers_u$ and, if an entry exists for v, the associated timeout in order to know whether a short preamble could be adopted or not.

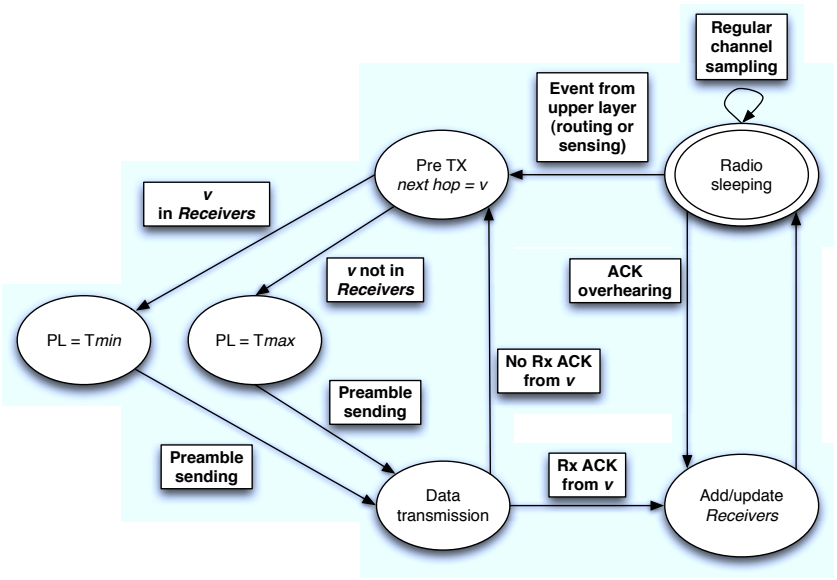


Figure 50: Transition-state machine of our auto-adaptive algorithm on the sender side.

The whole solution on the sender side is detailed in Figure 50. On this figure, *No Rx ACK* refers to the situation where v could not receive correctly the data, or the acknowledgment sent back by v was lost. For the sake of clarity, the limited number of packet retransmissions was not represented in this transition-state machine.

Considering acknowledgments, we also took into account the overhearing of these messages in order to enhance our solution. We implemented the fact that, for every acknowledgment sent by a node v to a node w, and overheard by a node u, then u can decide to add v to $Receivers_u$. By this way, if u sends data to v before the end of the associated timeout, a short preamble can be used. Note that v does not need to know about the adoption of a short preamble by u: the acknowledgment notifies that v has switched to a short SP, hence it will be able to synchronize with the short PL of u.

7.1.3 *Operations on the receiver side*

The procedure is detailed through Figure 51. Receivers do not have to maintain a list of senders. Let a node v configured with SP = T_{max}. The first time a node u sends data to v, v acknowledges the reception (noted as *Data OK* in the figure) and agrees on a short preamble, thus changing SP to T_{min}. The time period during which v will remain in this state is noted as $Timeout_v$ (its value is the same as the one used for the timeouts on the sender side). This parameter is further discussed in Section 7.3. Every new data reception resets the ongoing timeout to $Timeout_v$, thus extending the period during which v keeps an SP equal to T_{min}. In case v receives data destined to another sensor node or if the data is incorrect or incomplete (noted as *Data error* in the figure), it does not send any acknowledgment and goes back to the state it initially came from.

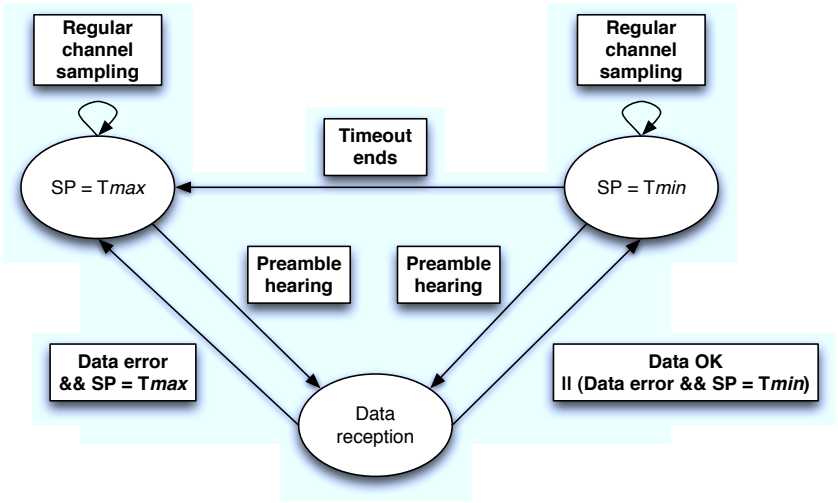


Figure 51: Transition-state machine of our auto-adaptive algorithm on the receiver side.

7.1.4 *Overall operations*

A general overview of our proposal is depicted in Figure 52. The *sender* node only modifies its PL value upon an acknowledgment received from the *forwarder* node. It will then be able to send the next data packet using a short PL. The forwarder node modifies its SP value to the minimum one upon reception of the first data packet from the sender node. This allows him to synchronize correctly for the reception of the second data packet. When forwarding the first packet, it however still uses a long preamble. It will reduce the preamble size only from the second forwarded packet (not displayed in the figure). The *receiver* node receives the first forwarded packet, acknowledges it and switch its SP value in

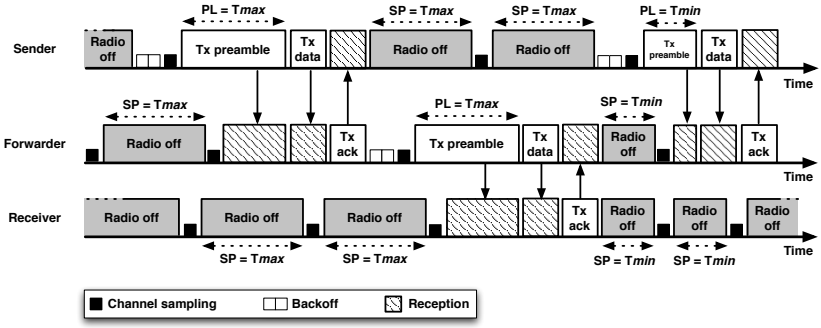


Figure 52: Overview of our auto-adaptive algorithm with one sender, one forwarder and one receiver.

order to be ready to hear the second packet on time. According to the role of the node (sender only, forwarder, receiver only), nodes modify either their PL value, or their SP value, or both.

This solution is also resistant to loss of messages that inevitably occur in real radio environments. If data packets are not received, then retransmissions occur until the limit is reached, leading to packet drop. In the case where acknowledgments are lost, the receiver has changed its SP to T_{min} while the sender keeps on sending a preamble of length T_{max}. Such situations can not lead to link-layer disconnections: the SP being shorter than the PL, the receiving node will inevitably hear the preamble. It simply wastes some energy that could have been saved with a shorter preamble.

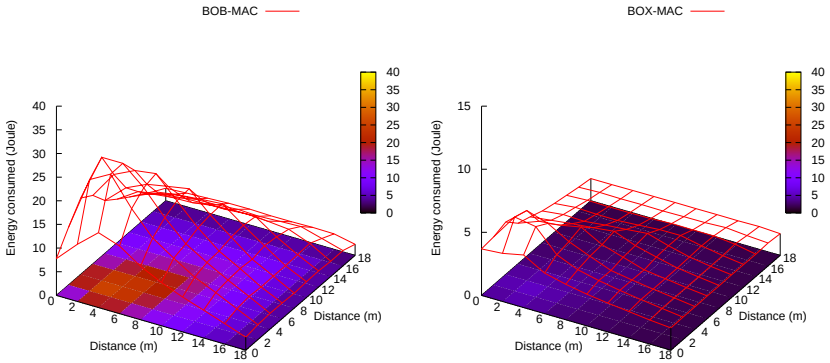
7.2 EVALUATION OF OUR CONTRIBUTION

We have performed a similar experiment with WSNet than the one presented in Chapter 6 (Section 6.1). We implemented our auto-adaptive algorithm as a new module that can be employed with the B-MAC or X-MAC implementations for WSNet. In the below results, the combination of our algorithm with B-MAC and X-MAC will be called Burst-Oriented B-MAC (BOB-MAC) and Burst-Oriented X-MAC (BOX-MAC) respectively.

We set T_{min} to 100 ms and T_{max} to 500 ms. With these values, we will get the benefits of a short PL in active places and a long SP in idle places. We chose a Timeout value of 10 seconds. Our algorithm remains efficient as long as Timeout is greater than the delay between two consecutive uses of an EE link, which depends on the routing policy. We use a random geographic routing (see Table 11 in Section 6.1), where a node forwarding a packet selects its next hop randomly among the neighbors that are strictly nearer from the destination than itself. Setting Timeout to the length of the event thus guarantees that EE links will remain established during the whole burst transmission.

The results exposed in the below Sections exhibit the benefits of our solution in terms of energy consumption, end-to-end and one-hop delays as well as packet losses.

7.2.1 *Energy consumption*



(b) (c)

Figure 53: Map of the energy consumption in the sensor grid at the end of the simulation for (a) BOB-MAC and (b) BOX-MAC.

Figure 53 depicts a map of the energy consumption of our sensor grid at the end of the simulation for BOB-MAC and BOX-MAC. The results for BOB-MAC are pretty similar to those exposed for B-MAC with a 250 ms LPL in Section 6.2.1. Table 17 confirms that the former consumed around 975 J in the whole network while the latter presented a depletion of 972 J. The repartition of the consumption is however slightly different: BOB-MAC consumes less energy in radio initialization (thanks to the use of long preambles on idle nodes) but more on reception (due to the seldom use of such preambles, that are overheard by multiple neighbors).

MAC protocol	*Total energy consumed in the network (Joules)*	*Main reasons (% of total consumed)*			
		Tx	*Rx*	*Idle*	*Radio init.*
BOB-MAC	975.66 (± 9.41)	21.97	56.88	5.3	15.85
B-MAC (LPL 250 ms)	972.22 (± 6.29)	22.91	42.07	7.54	27.49
BOX-MAC	261.42 (± 0.98)	14.25	4.57	19.34	61.85
X-MAC (LPL 500 ms)	380.48 (± 3.19)	30.62	15.22	19.11	35.04

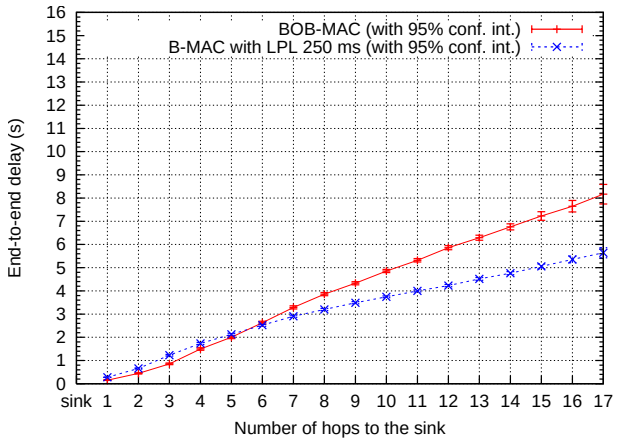
Table 17: Average energy spent in the network for BOB-MAC and BOX-MAC at the end of the simulation (with 95% confidence interval). For reference, the best results obtained by the plain B-MAC and X-MAC protocols are also given.

BOX-MAC takes great benefits from our solution. The energy consumption is significantly reduced on the routing path. Network-wide, it uses 31% less energy than when X-MAC is used with a LPL pre-configured to 500 ms (261 J versus 380 J). With BOX-MAC, the experiment could last around 1315 days before all nodes would die (against 900 days without our algorithm), and the most consuming node would die after 586 days (against 280 days previously).

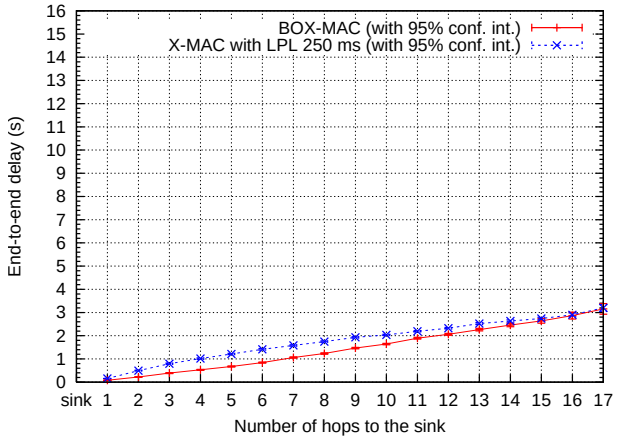
7.2.2 *End-to-end and one-hop delays*

The average end-to-end delay according to the number of hops to the sink is depicted in Figure 54 for BOB-MAC and BOX-MAC.

BOB-MAC achieves a little worse than when B-MAC is configured with a fixed LPL value of 250 ms. This can be explained by the



(a)



(b)

Figure 54: End-to-end delay according to the number of hops to the sink for (a) BOB-MAC and (b) BOX-MAC. Approaching results for plain B-MAC and X-MAC are given as reference.

one-hop delay which is in average 311.2 ms (± 2.34) with our algorithm while it was about 288 ms in the pre-configured B-MAC case. Although we could expect a lower medium access delay, this average value can only be reduced if enough short preambles are sent successively to make the initial long one of 500 ms profitable. It thus becomes obvious that the longer an event lasts (i.e. the more data a node sends in burst), the lower the average one-hop delay will be. As a side note, throughout the whole experiment 63% of the preambles sent by nodes were short ones. This low value can be explained by the fact that EE links are not used at their best capacity. Indeed, the random geographic routing model opens multiple routing paths at each forwarding node, which requires new EE link negotiations hence long preambles. We will discuss a way to solve this issue in Section 7.3.1.

BOX-MAC achieves roughly the same node-to-sink delay as when X-MAC is configured with a fixed LPL value of 250 ms. With an average one-hop delay of 137.6 ms (± 1.23), BOX-MAC can handle burst transmissions without provoking congestions. 49% of the preamble sent during the whole experiment are short ones (i.e. initially configured to last a maximum of 100 ms).

7.2.3 *Packet losses*

Losses at the sink are exposed in Table 18. BOB-MAC presents as much losses as when it is pre-configured with a LPL of 500 ms. On the contrary, BOX-MAC achieves good results: it performs better than if it were configured with a LPL of 250 ms. In order to better understand why losses with BOB-MAC are that high, we expose in Table 19 the main reasons of the losses at the radio level.

We can note an increase in the proportion of data packets not captured and packet errors for the experiment running BOB-MAC. Actually, a weakness in our protocol makes it more sensible to the hidden node problem. When a sender u transmits a long preamble of 500 ms (which happens in 37% of the cases, as stated in the previous Section), it may collide up to five times with short

MAC protocol	*Packet sent by applications*	*Packet received at the sink*	*% of losses at the sink*
BOB-MAC	1800	1281.6 (± 13.45)	28.80
B-MAC (LPL 500 ms)	1800	1289.65 (± 13.48)	28.35
BOX-MAC	1800	1722.75 (± 6.51)	4.29
X-MAC (LPL 250 ms)	1800	1691.45 (± 11.56)	6.03

Table 18: Average packet losses at the sink (with 95% confidence interval) for BOB-MAC and BOX-MAC. As reference, approaching results obtained by plain B-MAC and X-MAC are also given.

MAC protocol	*Packets forwarded by the MAC*	*Packets received at the MAC*	*% of losses*	*Main reasons (%)*		
				Radio off	*Packet not captured*	*Packet error*
BOB-MAC	16805.6	11500.45	31.5	46.6	31.3	21.5
BOX-MAC	16572.0	15074.35	9.0	50.0	8.5	38.7

Table 19: Average losses at the MAC layer and main reasons for BOB-MAC and BOX-MAC. Main reasons are given as a percentage of the total of packet losses.

preambles of 100 ms sent by another transmitter u'. The data packet sent by u' may thus reach the maximum retransmission threshold (set to 3 in our simulations). The receiver node v located between both senders will thus not be able to receive correctly the data packet sent by u'. Multiple solutions could help us to decrease the losses. We could increase the number of retransmissions or use RTS/CTS messages, but both would also increase the energy consumption on sender nodes. Another possibility would be to try to increase the percentage of short preambles sent in the network. This would immediately inhibit the hidden node problem that we experience. A way to achieve this would be to reduce the route diversity, as we already discussed in Section 7.2.2. We will further investigate this solution in Section 7.3.1.

We can note that BOX-MAC is much less impacted by this problem. Only 4.3% of the data packets do not reach the sink. Although 51% of the preambles sent during the whole experiment are long ones, they actually do not last 500 ms thanks to the early acknowledgment feature inherited from X-MAC. Coupled with a retransmission threshold set to three, it seems resilient to losses.

7.2.4 *Summary*

Our algorithm to adapt the LPL performs differently when combined with B-MAC and X-MAC. BOX-MAC gives encouraging results. In terms of energy, we outperform all the experiments where X-MAC is used pre-configured. The end-to-end delay is roughly the same as in the 250 ms LPL case. With less than 5% of packet losses, it proves that energy savings are not accomplished to the cost of reliability.

BOB-MAC exhibits results that are globally worse than when B-MAC is configured with a 250 ms LPL. Although energy consumptions is kept to a similar level, end-to-end delays and packet losses are worse. We have however spotted several reasons for these flaws. We believe that our solution could be improved by the mean of small information exchanged with the routing layer. We will further expose these ideas in the next Section.

7.3 ENVISIONED IMPROVEMENTS

We have briefly introduced in the previous Section several possible optimizations for our solution. We now detail three enhancements that could further improve our algorithm by increasing the proportion of small preambles sent in the network. Along with these proposals, we also expose preliminary results that will help to understand the possible benefits.

7.3.1 *Cross-layer optimizations*

Once EE links are established, nodes should try to use them as much as possible. The utilization of a given communication link is however imposed by the routing policy. We thus think about two possible improvements:

- *Routing to MAC optimization:* our solution would work even better if the routing policy aimed at using the same next-hop, or at least the same set (as small as possible) for burst transmissions. This would mean long-term established short preamble links (see ① on Figure 55),
- *MAC to Routing optimization:* a reverse optimization would be to have the next-hop nodes of u elected at the routing layer on the basis of the $Receivers_u$ list. Indeed, nodes belonging to $Receivers_u$ are already identified as being configured with a SP value of T_{min}, thus meaning energy-efficient transmissions with short preambles (see ② on Figure 55).

We have implemented the first proposal in the WSNet simulator, and present preliminary results hereinafter. Basically, we have maintained a random geographic routing model, but the next

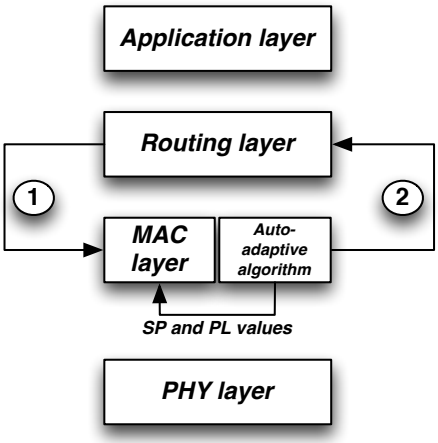


Figure 55: Multiple opportunities for further cross-layer optimizations of our algorithm.

hop is randomized only every X forwarded packets. In our case, we have used $X = 10$ in order to match the number of packets sent by the application layer upon an event. The most interesting results are experienced with BOB-MAC. We depict the energy consumption in the network in Figure 56. We can see that it performs better than any other experiments with a pre-configured B-MAC. The network-wide consumption is reduced to 611.89 J (while it was 975 J without the routing optimization). The one-hop delay is decreased to 187.8 ms (with 84.7% of the sent preambles being short), which allows to reduce the packet losses at the sink to an average of 21.5%. We plan to implement other optimizations in order to decrease these losses to a negligible value.

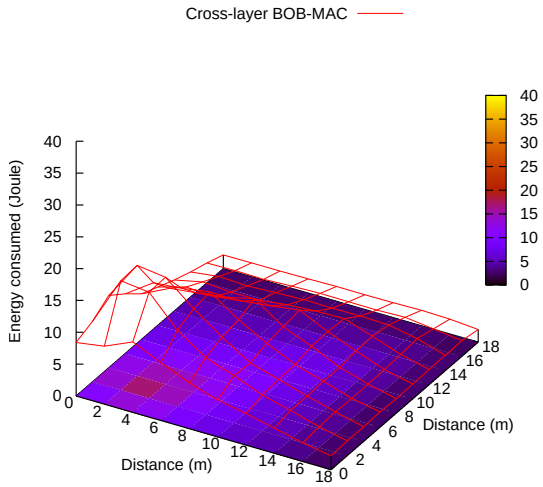


Figure 56: Map of the energy consumption in the sensor grid at the end of the simulation for cross-layer BOB-MAC.

Similarly, we obtain even better results when cross-layer information is used in combination with BOX-MAC. As an example, the network-wide energy consumption is decreased to 229 J, and the one-hop delay is reduced to 104.8 ms (with 84.3% of the sent preamble being short). Packet losses at the sink are brought down to 2.9%. Of course, such results suppose that one knows a-priori the number of packets that would be sent consecutively as a burst by the application layer. The value of X could be statistically computed over time in the network, for example.

7.3.2 *Tuning EE links default duration*

Future work will also focus on an adaptive timeout function optimization. Indeed, once a receiver v has changed SP into T_{min}, not receiving any data before the end of $Timeout_v$ will result in v changing SP back to T_{max}, thus annihilating our solution. The duration $Timeout_v$ could be made longer to solve this and allow nodes to take advantage of the EE link for a longer period. Yet, a

too long timeout leads to energy waste as nodes would wake up more regularly.

Far from being trivial, a default timeout may actually depend on several parameters. For instance, we set it to 10 seconds in our experiments. Such a short period reduces the benefits of our solution as soon as data packets are forwarded through the same routing path at intervals larger than 10 seconds. In a more adaptive approach, this timeout could also be auto-adapted. For instance, if a sender node u learns about further transmissions from the application layer (for instance, n messages will be sent during the next m seconds), it could inform v during the data transmission, which would as a result keep a short SP for this amount of time. The data acknowledgment would also include this time period to confirm to u (and potential neighbors) the timeout value that should be stored in $Receivers_u$.

7.4 PRELIMINARY IMPLEMENTATION ON TINYOS

We are currently implementing our algorithm on TinyOS [25]. As an example, we operate it between 2 TelosB motes [24] (one sender and one receiver) and depict their radio status in Figures 57 and 58. Initially, both nodes operate with PL = SP = 500 ms. The sender emits one data packet every 500 ms from t = 3000 ms to t = 10000 ms. Figure 57 shows the sender (upper graph) using a long preamble (PL = 500 ms) for the first data transmission. Upon reception, the receiver (bottom graph) switches to a short SP (100 ms), which guarantees the reception of the next short preambles transmitted by the sender.

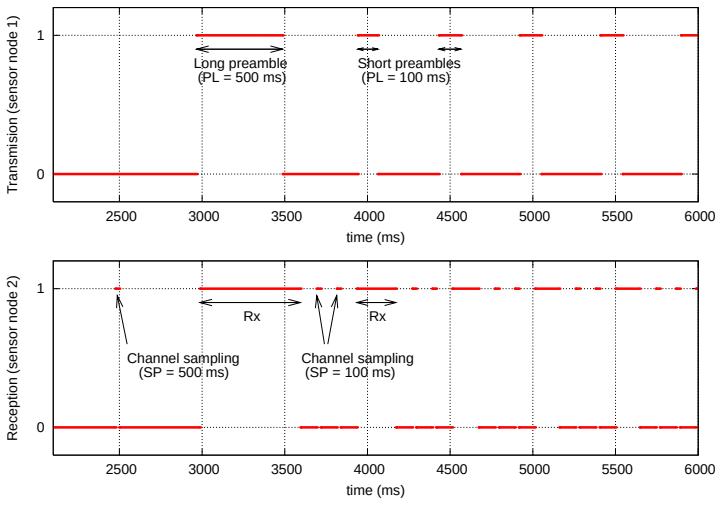


Figure 57: BOB-MAC on TinyOS: the sender and receiver agree on a short PL upon the first data transmission.

Figure 58 confirms that once the burst transmission is completed, the receiver falls back to a long SP upon expiration of the timeout.

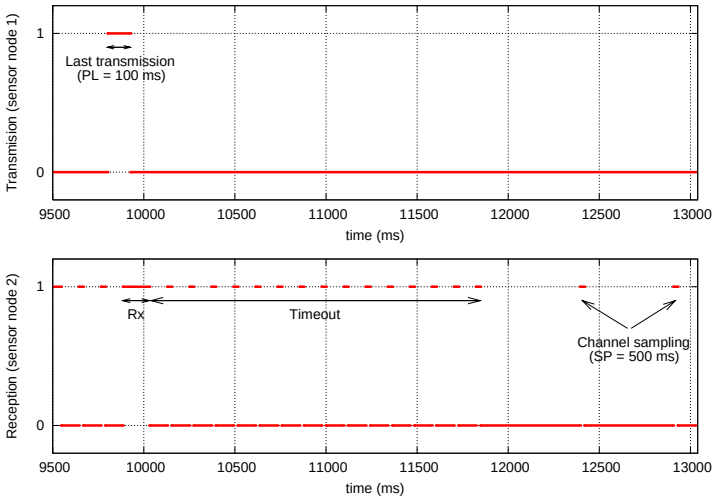


Figure 58: BOB-MAC on TinyOS: the receiver switches back to a long SP once the timeout expires.

7.5 CONCLUSION

The B-MAC and X-MAC sampling protocols propose a set of interfaces to easily configure them according to the constraints of the network. While it makes these protocols a likely solution for a number of deployments, it however requires a certain expertise. Furthermore, these versatile protocols must be used with caution as performing a reconfiguration while the network is running could lead to synchronization issues among the nodes. So far, these protocols have thus been pre-configured prior to the deployment.

In the future, we can expect WSN deployments to be performed at large scale, and to last months without operations of any kind. The scenario we have exploited in this contribution also assumed that such deployments would have to monitor events that happen at an unpredictable location. This is a likely assumption if we contemplate mobile scenarios or detection tracking applications at large scale. The resulting burst transmissions require different medium access policies to remain energy-efficient while guaranteeing good performances in terms of delays and packet losses. It is thus important to develop auto-adaptive protocols that could adjust to such constraints.

By automatically adapting both the Preamble Length (PL) and the Sampling Period (SP) of the LPL mechanism, an optimum value could be used on the routing paths as well as idle places. This is what we tried to achieve by proposing an auto-adaptive

configuration algorithm. It uses distinct values for SP and PL while guaranteeing link-layer connectivity among the nodes. Our algorithm has the key advantage to lower the required expertise for a user to configure the MAC layer. For a given deployment, the configurable parameters are basically the values between which PL and SP can switch (T_{min} and T_{max}), and the duration of energy-efficient links ($Timeout$).

BOX-MAC, an implementation of our contribution on X-MAC, has proven its energy-efficiency without deteriorating the reliability. In terms of energy, BOX-MAC outperforms all the experiments where X-MAC is used pre-configured. The end-to-end delay is roughly the same as with a fixed 250 ms LPL. With less than 5% of packet losses, it proves that energy savings are not accomplished to the cost of reliability. We could however spot several flaws when we combined our algorithm with B-MAC. Especially, our solution is sensible to the routing model used on the nodes. In order to get even better performances, one lead could be to use cross-layer informations between the routing and MAC layers, which would make the establishment of the energy-efficient links more profitable. As briefly exposed, our solution would work even better if the routing policy aimed at using the same next-hop, or at least the same set (as small as possible) for burst transmissions. As a result, the $Timeout$ value could be computed and optimized by each node according to its route diversity.

Note that our contribution could be used in combination with other preamble sampling protocols as well, such as MFP-MAC [11] or WUF [88] for example. We also believe that it could be adapted to some hybrid protocols that use a preamble in their active period, as performed by SCP [111].

GENERAL CONCLUSION AND PERSPECTIVES

GENERAL CONCLUSION

In the last decade, Wireless Sensor Networks (WSNs) have progressively changed the way to monitor environmental and physical phenomenons. The deployment of tiny embedded devices, capable of operating unattended for long periods of time, enables the collection of information in a less intrusive fashion. The collected data often helps scientists to better understand the observed scenes in space and time. Multiple successful deployments have already demonstrated the faculty of WSNs to deal with habitat, environmental or structural health monitoring, surveillance systems and home automation. The energy constraint inherent to the battery-powered sensor nodes has motivated the research community to define new energy-efficient protocols. The radio transceiver being identified as the main source of energy dissipation, the Medium Access Control (MAC) layer has been the field of numerous optimizations. Especially, new communication schemes have emerged in order to put the radio into the sleep mode as much as possible while avoiding deafness among the sensor nodes.

More recently, the appealing promises brought by the use of mobile sensors has led researchers to investigate the ability of these schemes to operate in dynamic networks. Most of the MAC protocols for WSNs present good performances for a pre-defined scenario, but reveal multiple issues when confronted to evolutions in the topology or the traffic pattern. Multiple optimizations have been proposed in order to better integrate the mobile sensors in the communication scheduling of their neighborhood, as well as adapt to time-varying traffic loads in the network. Still, these solutions can hardly operate in large-scale and dense networks, and often suffer from a high control overhead. In this thesis, we have detailed how network dynamics, and more particularly mobility and auto-adaptation to the traffic, could be addressed in an energy-efficient manner at large scale. For that purpose, we have based our contributions on preamble sampling schemes, which can operate in a distributed manner without the need for time synchronization. The flexibility they offer makes them more suitable to deal with dynamic scenarios with great number of sensor nodes. Still, their performances can be affected by a number of shortcomings that our contributions aim at reducing.

We have identified in Chapter 3 several issues that a mobile node may face when operating a preamble sampling protocol in a dense environment. Especially, mobile sensors experience packet losses due to congestion and synchronization issues. In order to mitigate these problems, we have proposed to let a mobile node take possession of the medium owned by a fixed sensor node. This has led to the definition of two new MAC

protocols, *Machiavel* in Chapter 4 and *X-Machiavel* in Chapter 5. These protocols combine our proposal with a non-optimized and an optimized preamble sampling protocol (B-MAC and X-MAC respectively) in order to demonstrate that it can improve the performances of the protocols on which our contribution relies. In particular, our protocols perform significant reductions in packet losses and end-to-end delays in dense networks, without impairing energy-efficiency. With X-Machiavel, we have especially focused on a scenario where the mobile nodes numerically exceed the fixed ones, which has been hardly addressed in the literature so far. With a ratio of mobile to fixed nodes equivalent to 4, X-Machiavel divides the packet losses and the medium access delay of the mobile nodes by a factor 2 compared to X-MAC. The 33% decrease in the overall energy consumption attests that X-Machiavel does not trade-off energy to achieve these results.

The versatile features offered by some of the preamble sampling protocols makes them very suitable to various scenarios. This however require a certain expertise to pre-configure the MAC layer according to the characteristics of the deployments. Especially, the Preamble Length (PL) and the Sampling Period (SP) are setup with great care in order to sustain the expected traffic. Still, we have demonstrated in Chapter 6 that no perfect configurations exist when we operate preamble sampling protocols in a network with antagonist traffic patterns. Instead, the ability to reconfigure these parameters while the network is running could greatly improve the overall performances of the deployed WSN. For that purpose, we have proposed an auto-adaptive algorithm in Chapter 7 that targets energy-efficient burst transmissions. Combined with a preamble sampling protocol, it can dynamically configure suitable PL and SP values according to the load sustained by the node. Decisions are taken individually on each sensor node while avoiding deafness. Our proposal does not require any additional control messages, hence avoiding energy consuming transmissions. BOX-MAC, an implementation of our contribution on X-MAC, has proven its energy-efficiency without deteriorating reliability: network-wide, it drains 31% less energy than X-MAC and achieves a packet loss rate below 5% at the sink. We envision cross-layer optimizations between the routing and MAC layers in order to achieve even better performances. As an example, we could reach a 40% decrease in the energy consumption as well as a packet loss rate below 3% when BOX-MAC interacts with the routing layer.

We could observe that a very small number of real deployments have used MAC layers from the literature. Among the protocols overviewed in Chapter 1, their usage was limited to S-MAC in [18], B-MAC in [93, 99] and T-MAC in [58]. Besides, authors of [58] have discussed the difficulty to use solutions proposed in

the literature. In order to ease the adoption of our contributions, we have tried to design simple paradigms that could be applied to already existing preamble sampling protocols. Our systematic evaluation of our contributions combined to B-MAC and X-MAC has demonstrated the benefits of our proposal compared to the original protocols. Furthermore, we believe that the solutions presented in this thesis can be easily adapted to other preamble sampling protocols.

PERSPECTIVES

To our minds, implementation and empirical evaluation of our contributions is an important step to guarantee their robustness and efficiency at large scale. We are currently implementing our proposals on TinyOS, and we would like to validate this work by conducting an experiment on the Senslab platform [87]. Senslab is composed of 1024 sensor nodes distributed in 4 heterogenous platforms, each one proposing its own characteristics (mobile sensors and different radio chipsets) and topology (3D mesh and grid). We also would like to study the combination of Machiavel or X-Machiavel with our auto-adaptive algorithm, especially what could be the impact of using different PL and SP values on a mobile node. We expect a few flaws that could require some refinements to integrate them into a single protocol.

As briefly overviewed in Chapter 7, further optimizations of the MAC lies in the ability to exchange information between layers. Although we have tried to keep our contributions independent from the routing layer, it appeared that more energy savings could be performed with MAC to routing and routing to MAC optimizations. We would like to investigate this lead by improving our contributions with a set of cross-layer interfaces. The information received through them would be used as an input of our auto-adaptive algorithm. Hints on the expected traffic (from the application layer), the appearance or removal of routes (from the routing layer), or the Link Quality Indicator (LQI, from the physical layer) could govern the decisions taken by our algorithm at the MAC layer.

Studying layer-2 mobility in WSNs as well as layer-3 IPv6 mobility in traditional wireless networks (see Appendix A) questions us on the convergence of these two worlds. The use of IPv6 on tiny sensor nodes is now becoming a reality with the recent implementation of micro IPv6 stacks in embedded operating systems such as Contiki [28]. This great accomplishment will certainly be followed by other components of the IPv6 protocol suite. For example, an equivalent to the Network Mobility Basic Support (NEMO BS) protocol for WSN would enable transparent mobility of the sensor nodes between different WSN clouds. Both technologies are

already envisioned in the Intelligent Transportation System (ITS) area. On one side, WSNs appear as a likely solution for information collection in the communication system of vehicles. On the other side, IPv6 and NEMO BS have been adopted as the base of the Communication Access for Land Mobiles (CALM) architecture, which defines vehicle-to-infrastructure and vehicle-to-vehicle communication modes. Some of the ITS use cases, especially in the road safety field, could certainly benefit from the convergence of both worlds.

APPENDIX

A

OTHER CONTRIBUTIONS IN THE IPV6 WORLD

Contents

Rapid deployments of wireless technologies have stimulated the appearance of new types of user behavior and expectations. Lately, the concept of pervasive connectivity (i.e. being connected anywhere, anytime) became of crucial importance. The users expect to benefit from their usual network applications or services while on the move. However, layer-3 movements (i.e. when a roaming node moves across various IP networks) require the nodes to change their IP addresses to avoid routing issues. Without specific support, such IP address changes would break ongoing communication. In addition to the Mobile IPv6 (MIPv6) protocol [47] which is designed to enable host mobility across Internet Protocol version 6 (IPv6) networks, the IETF has defined the Network Mobility Basic Support (NEMO BS) protocol [27] which enables mobility of entire IPv6 networks. NEMO BS is especially expected to be deployed in the Intelligent Transportation System (ITS) to offer global and permanent IPv6 connectivity to the passengers [55].

As we will detail in Section A.1, an important feature of NEMO BS is the use of a Mobile Router (MR) that hides the mobility of the network to the hosts located behind this router. However, the MR symbolizes a single point of failure in the mobile network. For redundancy or load sharing purposes, it could be interesting to use multiple MRs in the same mobile network. After a brief

overview of the existing work in Section A.2, we will expose in Section A.3 how this could be achieved in a transparent manner for the hosts in the mobile network [53]. The evaluation presented in Section A.4 will detail the benefits of our contribution. Finally, conclusions and future work will be exposed in Section A.5.

A.1 CONTEXT

A.1.1 *The NEMO BS protocol*

While MIPv6 proposes a solution for host mobility, NEMO BS extends the protocol to support entire networks moving in the Internet topology. An overview of NEMO BS is given in Figure 59.

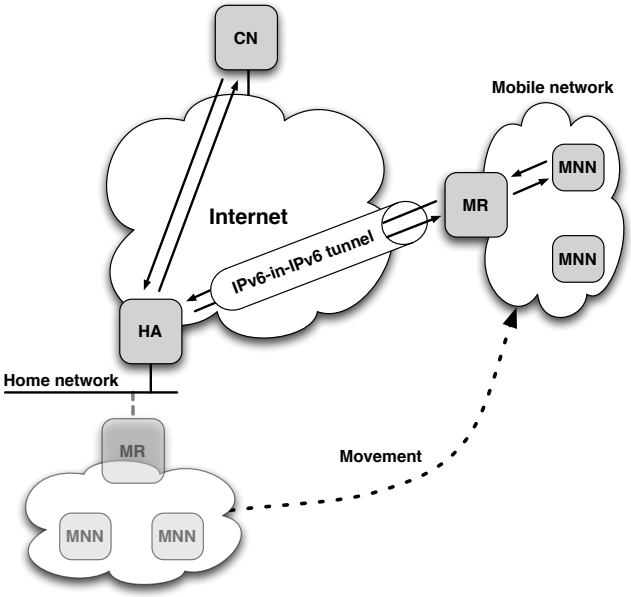


Figure 59: NEMO BS allows a whole network to move in the Internet without breaking ongoing communications.

Mobility is managed at the exit router (the Mobile Router, MR) of the moving network, and is transparent to all the nodes inside the network (called the Mobile Network Nodes, MNNs). The MNNs and the MR are always reachable at the same address while moving, thus all IP sessions are maintained even though the network changes its point of attachment in the Internet topology. This is possible thanks to a third entity, the Home Agent (HA), located in the home network of the MR and which maintains a binding between the Home Address (HoA) of the MR (which serves as an identifier), its Care-of Address (CoA) that represents its real location, and the IPv6 prefix advertised in the mobile network (the Mobile Network Prefix, MNP). This binding is updated every time the mobile network changes its point of attachment in the Internet topology. A bi-directional IPv6-over-IPv6 tunnel

is maintained between the HA and the MR, and all the traffic from a Correspondent Node (CN) located in the Internet toward the mobile network transits via the HA through this tunnel and conversely.

A.1.2 *Multi-homing with NEMO BS*

The MR may also be multi-homed, which means having several network interfaces connected at the same time or sequentially to the Internet. A node can get many benefits from multi-homing, such as fault-recovery. Using multiple interfaces at the same time is the most interesting scenario, as it also allows to distribute flows among the available interfaces. In a mobile environment, where connectivity may not be reliable all the time, the Multiple Care-of Addresses (MCoA) registration protocol [105] proposes an extension to NEMO BS to be able to use several interfaces at the same time on the MR (Figure 60). MCoA explains how the node can register several CoAs (each one identified by an unique number, the Binding Identifier or BID) at the same time (and for the same HoA) to its HA. This allows the MR to maintain multiple concurrent IPv6-over-IPv6 tunnels with its HA.

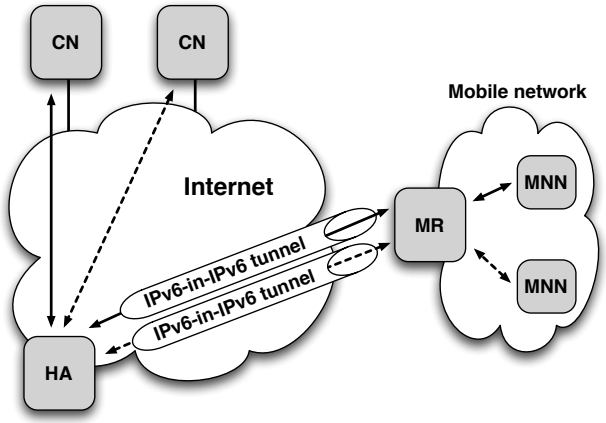


Figure 60: MCoA enables multiple paths between the MR and its HA.

Although these concurrent paths could be used as a backup for a main one, it would also be interesting to perform per flow path selection. It is typically needed when an application has specific performance requirements that makes one network interface more suitable than another. This implies that there must exist rules at both ends that determine for each packet which path should be used to transmit the traffic. For example, such rules could be described with the Augmented Backus-Naur Form (ABNF) language that we defined in [59] and transmitted between the MR and its HA with the exchange protocol specified in [91] or the one that we proposed in [35].

A.1.3 *Multiple Mobile Routers*

Without considering the HA, the reliability of the mobile network is directly dependent of the MR status. Whenever the MR fails (e.g. breakdown, lack of network coverage, etc.) the mobile network is disconnected from the Internet, hence disrupting all ongoing communication of the Mobile Network Nodes (MNNs). Although the lack of coverage of a particular wireless technology could be overcame with a multi-interfaced MR using the MCoA extension, the mobile network is still subject to a failure of the MR itself. In case of an intense traffic, data packets could be potentially delayed or even dropped when forwarded by the MR to the Internet. In fact, the MR may be a bottleneck for the whole mobile network, as the available bandwidth inside is usually greater than the one offered by the access networks the MR connects to.

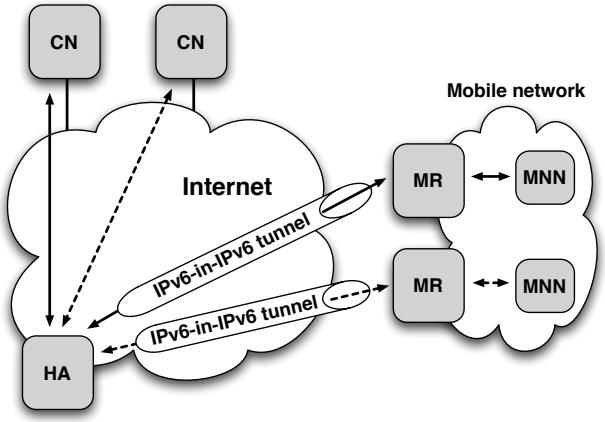


Figure 61: Multiple MRs with NEMO BS.

To address these issues, we could provide the mobile network with multiple MRs, as depicted in Figure 61. In addition to increasing the overall mobile network bandwidth and coverage, the cooperation of multiple MRs could enable a failover mechanism between MRs. Furthermore, the overall load of the mobile network could be shared between all available MRs, resulting in a more robust system. In Section A.3, we will present the first components of a new protocol which enables the simultaneous use of multiple MRs in the same mobile network. Based on Neighbor Discovery [72], our solution allows dynamic discovery between MRs, load sharing between all available MRs and failover support. Furthermore, our solution does not require additional software on MNNs so that the entire system remains compliant with legacy IPv6 clients.

A.2 RELATED WORK

In order to support the management of multiple MRs in a single mobile network, four points have to be considered. First, the MNNs have to select a default exit router among multiple ones. Then, all the MRs need to synchronize some information such as their availability and characteristics. Each MR also has to inform its HA about how the incoming traffic should be routed toward the potential multiple paths. Finally, the multiple HAs have to exchange information about the MR they manage. Research on multi-homing in NEMO BS is relatively recent and only few articles in the literature address the management of multiple MRs located in the same mobile subnet. Furthermore, most of them do not focus on all requirements suggested in [73].

A first proposal [22] presents a dynamic load sharing mechanism taking place on HAs. This proposal considers various MRs and HAs being held by several operators. Each MR in the mobile network can discover, authenticate and register a neighbor MR to its HA. Each HA can therefore maintain a list of alternative tunnels toward different HAs and MRs to reach its own MR. The HA can then perform load sharing between the legacy NEMO BS tunnel and alternative tunnels using tunnel latency as a metric. However the authors do not detail how the traffic is really redirected (packet by packet, flow by flow, etc.). In addition, it is fairly improbable that various operators accept to relay (through their HAs) traffic sent to clients belonging to a competitor.

Another proposal [23] presents a protocol focusing on reliability. In essence, the proposed solution allows a MR to act as a substitute for a failed one. Note that MRs may belong to different operators. Three error cases are supported: failure of the egress or ingress interfaces, and complete MR failure. Upon failure detection, a neighbor MR registers to the HA of the failed MR in order to provide Internet connectivity to the MNNs on behalf of the failed MR. Although this solution presents an interesting MR redundancy mechanism, it requires to register the MR from one operator to the HA of another operator.

Finally, the proposal presented in [100] allows a mobile network to be served through multiple MRs transparently to the MNNs. Basically, all the MNNs are connected to a unique MR, known as the primary MR, and the other MRs are seen as virtual interfaces of the primary MR. All traffic from MNNs is therefore sent to the primary MR which forwards it among all the available interfaces (real or virtual) according to the installed routing policies and preferences. However, the transmission of a packet through a virtual interface would generate an overhead in the mobile network as such packet is encapsulated by the primary MR before being sent again on the mobile subnet to a non-primary MR. Such overhead may

seriously degrade the quality of effective communications in case of a wireless mobile subnet.

On the other hand, there are several schemes addressing load sharing or redundancy between routers in fixed IPv6 networks such as [71]. However, these proposals are hardly suitable to mobile networks as they were not designed considering the problem in the mobility context.

A.3 OVERVIEW OF THE SOLUTION

Our proposal focuses on multiple MRs located in the same mobile subnet. These multiple MRs are associated to one or multiple HAs and advertise the same mobile network prefix in the mobile subnet. This scenario refers to the (n,*,1) model as described in [73]. Although the mobile network of a vehicle might be divided in multiple subnets, it is likely that the user segment (where the MNNs are located) will be served by MRs that belong to the same operator. We believe that this model is one of the most practical ones [36], all MRs and MNNs being on the same layer-2 link, i.e. each node (either MR or MNN) could directly join any other node in the subnet.

Our solution focuses on the assignment of a default router to the MNNs, i.e. through which router all the traffic from a MNN to the Internet is sent. We do not consider here how the downstream flows can follow the path back to that very router. We are currently extending the proposed solution to take this part into account. Note that we use advanced features of Neighbor Discovery, so readers may refer to [72] for further details on its related mechanisms.

A.3.1 *Discovery of neighbor MRs*

Before cooperating, all MRs of the mobile network have to discover their potential neighbor MRs. Each MR is pre-configured with a role: Master or Slave. The Master MR (which is unique in the mobile network) is in charge of the selection of the default MR for each MNN. In addition, the Master MR carries out the transmission of router advertisements over the mobile subnet. A Slave MR monitors the status of the Master and may react upon detection of Master MR failure (see Section A.3.3). A Slave MR could also be selected (by the Master) as the default router for a set of MNNs. Note that the roles could be assigned dynamically with a well-known and efficient election algorithm (e.g. the one defined in VRRP [71]) at system startup or upon failure of the Master.

A MR periodically announces its presence by sending several parameters over the mobile subnet. For doing so, we have defined a new ICMPv6 message referred to as Neighbor Router

Advertisement (NRA). We could assume to use existing router advertisements with new options instead, but in addition to their size limitation which restraints the number and the size of new options (ICMPv6 messages can not exceed the minimum IPv6 MTU), this would provide unsolicited information to MNNs. A NRA message includes (but is not necessarily limited to) the ingress interface link-layer address, the ingress interface link-local IPv6 address, the role and the currently used bandwidth ratio of the originator MR coupled with a lifetime. The bandwidth information is a ratio between the bandwidth currently consumed and the overall theoretical bandwidth available on the interface(s) that connects the MR to the Internet. The lifetime refers to the length of time that the provided information is valid. Note that NRAs are sent periodically to the all on-link IPv6 routers multicast address.

The Master MR maintains the status of every Slave MRs in a cache called the Mobile Router Status (MORS) cache. Each MR is registered together with all parameters provided in NRAs. The MORS cache is dynamically updated upon reception of a NRA. Entries from the MORS cache are automatically deleted once the lifetime that was previously announced in the NRA has expired. Whenever the Master MR has to select a default MR for a MNN, its decision is based on the information currently recorded in the MORS cache.

A.3.2 *Default MR selection*

Our targeted environment provides the mobile subnet with multiple MRs each directly connected to the Internet through its own interface(s). We therefore would like to benefit from these multiple paths to efficiently share the load between all MRs without modifications on the MNNs.

Each MR is statically configured with a virtual link-local IPv6 address which is shared among all MRs. This address is only used for communication between MRs and MNNs. All communications among MRs are achieved with their real and unique link-local IPv6 addresses. Note that only the Master MR defends this virtual address against stateless auto-configuration performed by another nodes [97]. The rationale behind the use of a virtual link-local IPv6 address is that we can not use a multicast or explicit anycast address as the IPv6 source address of router advertisements has to be a unicast link-local IPv6 address [72]. We could also assume to use the same link-layer address on each MR as suggested in [71], but in our proposal multiple routers are active at the same time. The vision of the same link-layer address on multiple ports simultaneously would be considered as a loop by the spanning tree protocol running on layer-2 devices.

In addition, each MR maintains the list of MNNs for which it acts as the default router. As previously mentioned, only the Master MR sends router advertisements over the mobile subnet. In these messages, the Master MR is configured to set the virtual address as the source address and to not set the source link-layer address option. Upon reception of such a message, a MNN could still achieve the IPv6 stateless address auto-configuration process but no new entries are created yet in the neighbor cache of the MNN (such configuration is defined by the specification of the Neighbor Discovery protocol [72]). As a reminder, the neighbor cache is a set of neighbor's on-link unicast IPv6 address and link-layer address tuples (with additional parameters) to which traffic has been sent recently.

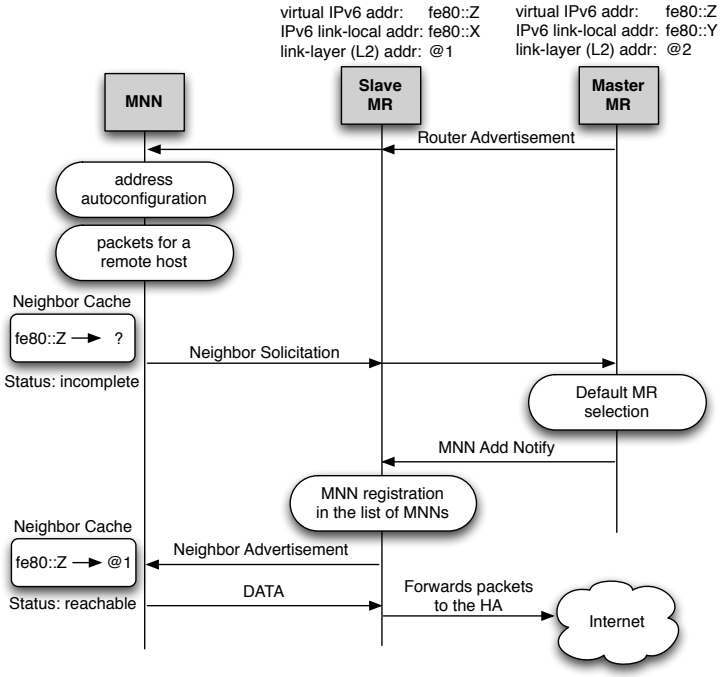


Figure 62: The default router selection for a MNN.

When a MNN wants to send its first data packet to a remote host, it first adds a new entry for its default router in its neighbor cache. The default router is referred to in the neighbor cache by the virtual link-local IPv6 address discovered during the IPv6 auto-configuration process. At this stage, this entry is in the *incomplete* state which means that the MNN has to resolve the link-layer address of the default router before being able to send a packet to this router. Following the standard procedure, this operation is initiated by sending a Neighbor Solicitation (NS). In our protocol, when a NS is sent to resolve the link-layer address of a router (i.e. the link-layer address associated to the virtual link-local IPv6 address), this message is only intercepted by the Master MR. Upon reception of such a message, the Master MR has to select,

among all available MRs, one to which delegate the MNN. This selection could be based on the bandwidth information currently announced by each MR and maintained in the MORS cache. If the selected MR is not the Master itself, we have defined a new message, known as MNN Add Notify (MAN), to notify a Slave MR about the delegation of a MNN. This message contains among others the IPv6 address of the targeted MNN. The MR to which the MNN has been delegated then adds the client in its list of MNNs and sends it back a Neighbor Advertisement (NA) with the target address configured to its link-layer address. When the MNN receives this message, it updates its neighbor cache (the entry corresponding to its default router goes to the *reachable* state) and starts to send its data packets to the delegated router. Figure 62 illustrates this procedure.

A.3.3 *Failure detection and node redirection*

Three cases have to be considered when designing a MR failover mechanism: ingress interface failure, egress interface failure and complete MR failure. The MNNs associated to a MR facing one of these failure cases have to be redirected to a functional MR. Our proposal considers the ingress interface and complete MR failure cases similarly as the failed MR is no longer reachable from the mobile subnet. Furthermore, the mechanisms presented here relies on Neighbor Discovery (the neighbor unreachability detection in particular) to redirect MNNs from one MR to another.

The ingress interface failure of a MR (or complete MR failure) is detected by a neighbor upon non-reception of NRAs whatever the failed MR is configured as Master or Slave. When a certain number of NRAs from the Master MR have been missed by the Slave, it assumes that the Master MR became unavailable. Missing a single NRA does not necessarily imply a Master MR failure as the event may be due to a link layer collision. Upon failure detection, a Slave MR should initiate the election of a new Master. Note that changing the Master MR while operating the protocol may be expensive as it includes election and potential MNN redirections. As a result, the future election mechanism should ensure that a failed Master coming back online would go to the Slave state. When a Slave MR fails, the corresponding entry in the MORS cache of the Master MR expires. The MNNs bound to a failed MR are automatically redirected to another MR thanks to the neighbor unreachability detection mechanism defined in Neighbor Discovery [72]. Without reachability confirmation, the entry corresponding to the failed MR should be deleted from the neighbor cache of all MNNs bound to the failed MR. This initiates again next-hop determination and address resolution as presented in Section A.3.2.

The case of an egress interface failure is managed in a slightly different manner as the failed MR is still able to communicate with the nodes located in the mobile subnet. Our protocol thus also enables explicit reallocations of MNNs from a MR to another. Note that this mechanism can also be used to react upon MR overload (e.g. when the MR attaches to a lower quality access network) and to share the load as fairly as possible (e.g. when a new MR connects to the mobile subnet). The idea lying behind our proposal is to dynamically update the link-layer address of the default router recorded in the neighbor cache of the MNNs.

In order to delegate a set of MNNs to another MR, a MR sends a new ICMPv6 *MNN Redirect* to the Master MR. This message includes the list of MNNs identified together with their link-layer addresses, their link-local IPv6 addresses and their estimated bandwidth requirements. We envision that such selection could be based on preferences such as redirecting first the MNNs which consume most of the bandwidth in order to limit the number of redirected MNNs. Upon reception, the Master MR tries to select a new default MR for each MNN listed in the MNN Redirect message. This selection could be achieved upon matches between the bandwidth requirement of a MNN and the currently available bandwidth on MRs. Once the Master MR has selected a new default MR for a MNN, it sends a MAN message to the selected MR with the *redirect flag* set. Upon reception, the selected MR proceeds with the same actions as presented in Section A.3.2. However, a node could only update the cached link-layer address of an existing neighbor cache entry upon reception of a NA with the target link-layer address option and the override flag set [72]. Therefore, the selected MR should set the override flag in every NA transmitted following the reception of a MAN with the redirect flag set. According to [72], a node receiving such a NA has to update the link-layer address in the corresponding neighbor cache entry with the one supplied in the target link-layer address option. Once the Master MR has processed the entire list of MNNs, it sends an ICMPv6 *MNN Redirect ACK* back to the overloaded MR. This message reports the redirection status for each MNN. The redirection status could be set to either *successful* or *unable*. The overloaded MR could therefore remove from its list of MNNs every client for which the redirection status is successful. If the MR is still overloaded, it could try to delegate again several MNNs.

Note that our protocol remains fully compliant with the neighbor unreachability detection procedure [72]. Although all Slave MRs discard multicast NS messages (which are only processed by the Master MR), a MR (either Master or Slave) has to reply to every unicast NS sent to its link-layer address.

A.4 EVALUATION OF THE SOLUTION

A.4.1 *Implementation overview*

We have implemented the proposed protocol for the GNU/Linux operating system as an userland tool using the Python programming language. This implementation is used in conjunction with UMIP [101], the NEMO BS implementation for the GNU/Linux operating system. Our work uses the Scapy6 packet manipulation program [16] to define the new protocol messages, send them on the medium, and catch that very traffic from the network. Although such an userland implementation might not give as good performance results as a kernel one, it gives interesting preliminary results that we present in the next Sections.

A.4.2 *The test platform*

Our test platform is composed of two GNU/Linux routers (representing the MRs), one playing the role of the Master, the other one being the Slave. Those two routers interconnect the same subnetwork to the Internet. This subnetwork represents the mobile network, where two MNNs are located. A CN is located in the Internet and is used as the communication endpoint of both MNNs. The whole platform is interconnected using Ethernet for the communication medium, as we want to ensure a reliable link to evaluate the behavior of our protocol

All of these nodes are running the GNU/Linux operating system with a 2.6.22-3 kernel. Both UMIP and our implementation are used on the Master and the Slave only. The MNNs do not run any other protocols than the IPv6 protocol suite that is delivered with the system.

A.4.3 *Scenario and results*

In all the following scenarios, one of the MR is pre-configured as the Master, the other one taking the role of the Slave. As previously mentioned, router advertisements are only sent by the Master. Results presented here are obtained by taking the most relevant ones among 10 runs of each scenario.

Load sharing among multiple MRs

In this scenario, two MNNs are located in the mobile subnet. Each MNN wants to send a 300 kbps UDP flow to the CN. Each MR connects to the Internet with a 400 kbps connectivity. At the beginning, the Master is the only available MR in the mobile subnet, and therefore should become the default MR for the MNNs

when they start communicating with the CN. Next, the Slave MR connects to the mobile network and starts sending NRAs in the mobile subnet.

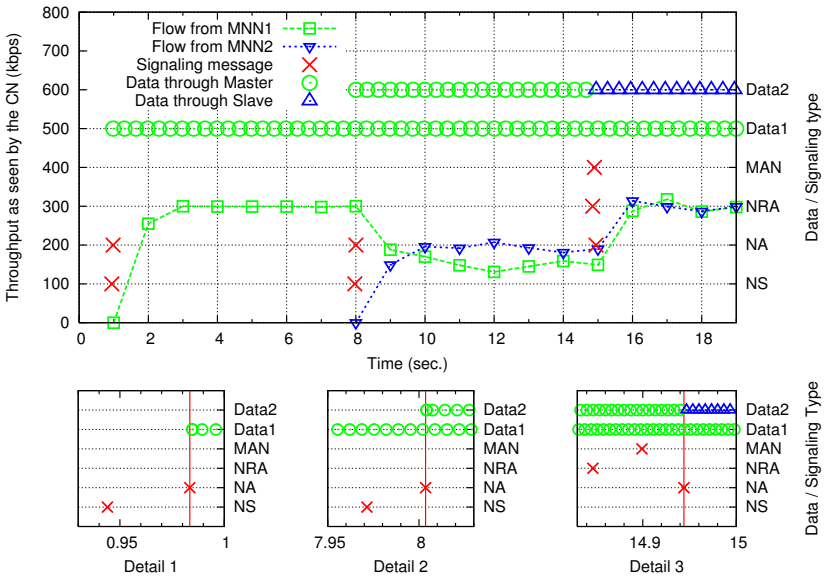


Figure 63: Load Sharing between Multiple MRs.

Figure 63 shows the whole experiment on the upper part, and various times around which an event occurs on the lower part. When the first MNN wants to communicate with the CN, it first tries to resolve the link-layer address of its default router by sending a multicast NS (see Detail 1). Upon reception, the Master adds the MNN to its list of MNNs and replies with a NA. Once the MNN receives this NA, it updates its neighbor cache accordingly and starts sending its data to the CN through the Master. These data packets are represented as *Data1* in the figure (only one packet over 50 is displayed in the upper graph for readability reasons). We can see that the CN starts receiving a flow of 300 kbps from the MNN. Later, the second MNN also wishes to initiate a communication with the CN. The same procedure as before applies (see Detail 2) as the Master MR is still the unique MR in the mobile network. When the second MNN also sends its data (represented as *Data2*) through the Master, we can see that the connectivity of the Master can not satisfy simultaneously the needs of both MNNs in terms of bandwidth. The upper graph shows that only 200 kbps of each flow is received at the CN which means that each flow experiences a significant degradation. Next, the second MR connects to the mobile network and advertises itself as Slave. Detail 3 shows that upon the reception of the NRA from the Slave, the Master tries to delegate one of the MNN to the Slave by sending a MAN message with the redirect flag set. Upon reception, the Slave accepts the redirection and takes care of updating the neighbor cache of the redirected MNN by sending

a NA to it with the override flag set. After the reception of this NA, we can see that all packets from the delegated MNN are sent via the Slave instead of the Master. The MNN has therefore correctly updated its neighbor cache with the link-layer of the Slave. The Master being no longer overloaded, the CN now properly receives the traffic from both MNNs as shown on the Figure. This illustrates how our protocol could dynamically distribute the overall bandwidth demand from the MNNs among the available MRs.

Dynamic redirection of a MNN from a Slave

In this second experiment, the default router that has been assigned to the MNN is the Slave. The MNN continuously sends an UDP flow to the CN (packets have an average size of 50 bytes and are sent every 10 ms). During that communication, the Slave explicitly asks the Master to reassign this MNN to another MR. Such query may happen for example if the Slave detects a failure on its interface connected to the Internet, or if it is overloaded and can no longer guarantee a good quality of service. Note that this scenario differs from the previous one as it illustrates the case of an overloaded Slave MR.

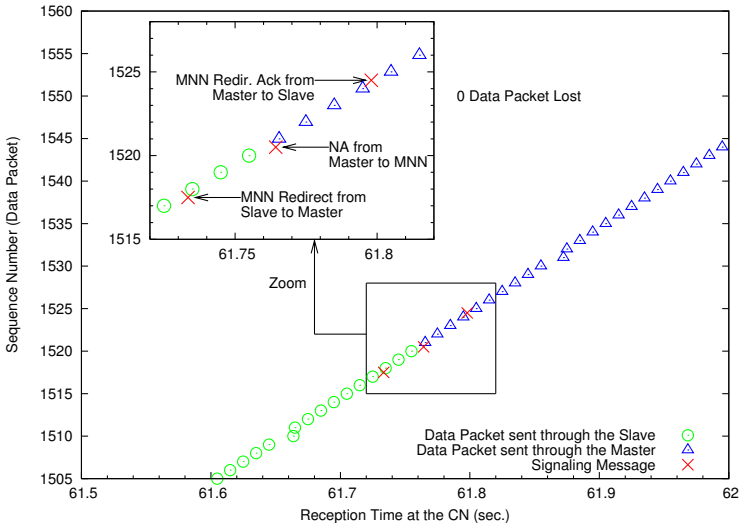


Figure 64: Dynamic Redirection of a MNN from a Slave.

The results are shown in Figure 64. Each dot represents the reception of a packet at the time indicated on the X-axis. The sequence numbers indicated on the Y-axis correspond to the data packet sequence numbers. At t = 61.735 s, we can see that the Slave MR initiates the redirection of the MNN by sending a MNN Redirect message to the Master. Upon reception, the Master selects itself as the default router for the MNN, then sends a NA (with the override flag set) approximately at t = 61.765 s to the MNN. From this time, all the packets from the MNN are sent through the Master: the link-layer address of the default router

of the MNN has been correctly updated in the MNN neighbor cache. The Master finally sends back to the Slave a MNN Redirect ACK message with a successful status code which completes the redirection procedure. We can underline the fact that no packets have been lost (or even delayed) during the transition. As a result, the default router change remains undetectable for the application on the MNN.

Failover upon router failure

In this last experiment, the initial configuration and MNN-CN communication pattern is the same as for the second one. This time, the network interface of the Slave connected to the mobile subnet fails without prior notice while the communication is ongoing.

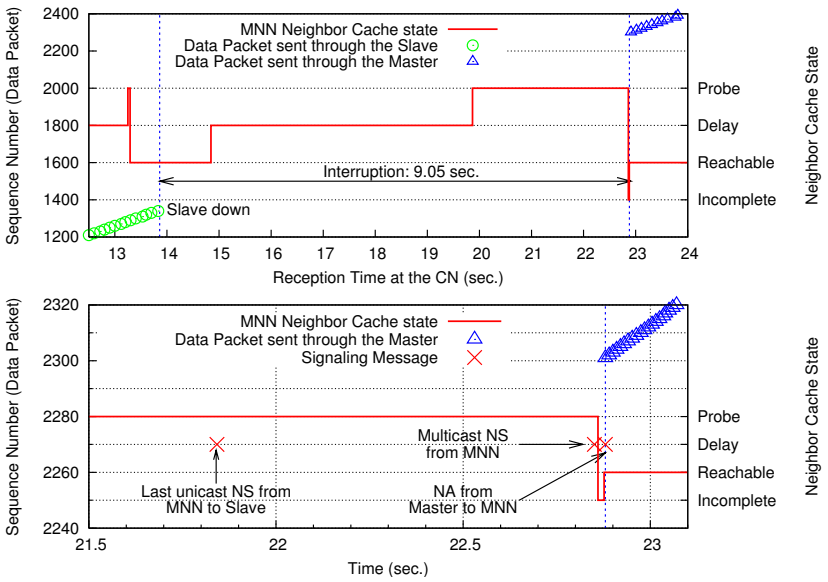


Figure 65: Failover upon MR failure.

The results are shown in Figure 65. The upper graph shows that the Slave fails at t = 13.9 s. It is not reachable anymore from the mobile subnet and thus can not either assure the routing of the data packets of the MNN. The states of the entry corresponding to the default router in the MNN neighbor cache reveals that after expiration of the *reachable* state, the MNN starts a neighbor unreachability detection procedure by entering the *delay* state and then the *probe* state. The lower graph of Figure 65 shows that after sending multiple unicast NS to the Slave without receiving any replies, the MNN fallbacks in the *incomplete* state (actually the neighbor cache entry for the default router is deleted but a new one is created right after because the MNN still has data to send). The MNN then immediately tries to resolve the link-layer address of its default router by sending a new multicast NS. In

the meantime, the entry corresponding to the Slave in the MORS cache of the Master has been deprecated at t = 14.5 s. When receiving the multicast NS, the Master assigns the MNN to itself (as it remains the unique MR in the mobile network) and replies a NA to update the neighbor cache of that MNN. Upon reception, the entry on the MNN goes directly to the *reachable* state, which triggers the transmission of the data packets to the CN, through the Master. In order to ensure that the Slave entry will be expired in the MORS cache of the Master before the MNN starts to send a new multicast NS, it is interesting to raise that the lifetime of a MORS entry should not exceed the duration of the *probe* state (fixed to 3 s in the Neighbor Discovery specification).

A.5 CONCLUSION

There are various points to take into account when considering the management of multiple mobile routers in the same mobile network. The solutions proposed so far to address the global problem hardly match all the requirements at the same time: they only consider either load sharing or mechanisms to react upon failure. With this contribution, we concentrate on the assignment of a default router to the mobile network nodes. Our solution proposes load sharing, dynamic redirection and failover mechanisms in order to fairly use all the resources available in the mobile network. Furthermore, the solution does not require any modifications on the client nodes located in the mobile subnet.

Through preliminary results presented in Section A.4, we have validated the behavior of our protocol and confirmed its accuracy with respect to the Neighbor Discovery protocol. Furthermore, explicit redirections of a mobile network node from a mobile router to another remain transparent for ongoing communications. Thanks to the neighbor unreachability detection, the nodes bound to a failed mobile router could automatically retrieve a functional mobile router. However, the default timer values defined for the neighbor unreachability detection may not match latency requirements of time-sensitive communication (we have experienced a flow interruption of approximately 9.05 s). We plan to reduce this disruption time by enabling the Master to send unsolicited NA upon failure detection of a Slave instead of changing default timer values. Encouraged by the results obtained for legacy GNU/Linux clients, we also have successfully experimented our protocol with Mac OSX and Windows XP clients. We are currently extending our contribution in order to consider how the downstream flows could use the same paths as the upstream ones.

B

THESIS' FRENCH VERSION

The next pages correspond to a summary of this thesis in French.

L'ACCÈS AU MÉDIUM FACE À LA DYNAMIQUE DES RÉSEAUX DE CAPTEURS SANS FIL

Contents

B.1 INTRODUCTION

Un réseau de capteurs sans fil (*Wireless Sensor Network*, WSN) consiste en une distribution spatiale d'équipements embarqués autonomes (des *capteurs*), qui coopèrent de manière à surveiller l'environnement de manière non-intrusive. Les données collectées par chaque capteur (tels que la température, des vibrations, des sons, des mouvements etc.) sont remontées de proche en proche vers un puits de collecte en utilisant des technologies de communication sans fil (Figure 66). Voilà une décennie que les contraintes inhérentes à ces réseaux attirent l'attention de la communauté scientifique. Ainsi, de nombreuses améliorations à différents niveaux de la pile de communication ont été proposées afin de relever les défis en termes d'économie d'énergie, de capacité de calcul et de contrainte mémoire imposés par l'utilisation d'équipements embarqués [112]. Plusieurs déploiements couronnés de succès démontrent l'intérêt grandissant pour cette technologie [84].

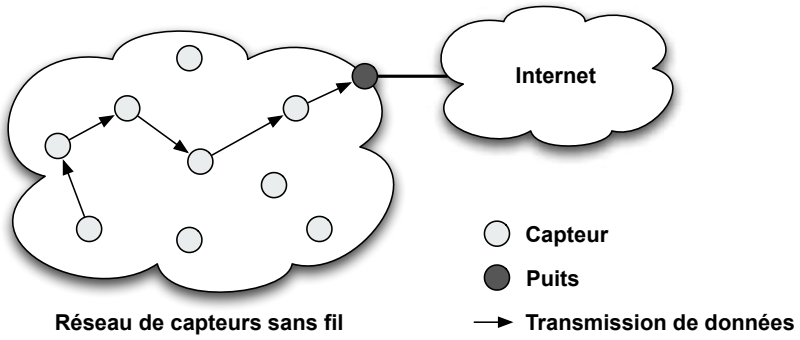


Figure 66: Un réseau de capteurs sans fil. Les données sont collectées par un capteur puis remontées de proche en proche vers un puits.

Les récentes avancées en termes d'intégration d'équipements et de protocoles de communication ont permis d'élaborer de nouveaux scénarios plus complexes. Ils mettent en scène des réseaux denses et dynamiques par l'utilisation de capteurs mobiles ou de différentes méthodes de collection de données. Par exemple, l'intérêt de la mobilité dans les WSNs est multiple dans la mesure où les capteurs mobiles peuvent notamment permettre d'étendre la couverture d'un réseau [114], d'améliorer ses performances de routage ou sa connexité globale [26]. Toutefois, ces scénarios apportent de nouveaux défis dans la conception de protocoles de communication.

Ces travaux de thèse s'intéressent donc à la problématique de la dynamique des WSNs, et plus particulièrement à ce que cela implique au niveau du contrôle de l'accès au médium (*Medium Access Control*, MAC). Après une présentation du contexte en Section B.2, nous exposerons dans la Section B.3 les limites des protocoles actuels. Nous présenterons ensuite nos contributions : la Section B.4.1 détaillera deux nouvelles méthodes d'accès au médium (*Machiavel* et *X-Machiavel*) qui permettent d'améliorer les conditions d'accès au canal pour les capteurs mobiles dans les réseaux denses. Notre deuxième contribution présentée en Section B.4.2 est un algorithme d'auto-adaptation destiné aux protocoles par échantillonnage. Il vise à minimiser la consommation énergétique globale dans les réseaux caractérisés par des modèles de trafic antagonistes, en obtenant une configuration optimale sur chaque capteur. Ce mécanisme est particulièrement efficace en énergie pendant les transmissions par rafales qui peuvent survenir dans de tels réseaux dynamiques. Nous conclurons ces travaux en Section B.5.

B.2 CONTEXTE DE RECHERCHE

Les contraintes imposées par les réseaux de capteurs sans fil ont poussé la communauté scientifique à proposer de nouvelles solutions d'accès au médium. Notamment, la réalisation d'une méthode de communication à faible coût énergétique a été jusqu'à présent l'un des défis les plus importants dans les WSNs [7] (comparé à la réduction des délais ou à l'équité dans les réseaux sans fil traditionnels). Les collisions, l'écoute involontaire (*overhearing*), l'écoute passive (*idle listening*) et l'utilisation de données de contrôle ont été identifiées comme sources principales de consommation d'énergie [7]. De nombreuses propositions ont ainsi vu le jour pour résoudre ces problèmes. Parmi les différentes fonctions d'une couche MAC, l'ordonnancement a notamment été le domaine de nombreuses améliorations. L'idée principale est d'éteindre la radio autant que possible tout en assurant la connectivité entre les capteurs. Pour satisfaire cela, deux catégories majeures de protocoles ont vu le jour : les protocoles par échantillonnage et les protocoles synchronisés.

Les protocoles par échantillonnage tels que B-MAC [77] (Figure 67) utilisent l'envoi d'un préambule avant les données. Chaque capteur présent dans le réseau allume périodiquement sa radio et écoute le médium. Si aucun signal n'est détecté, le noeud éteint sa radio. Si en revanche un préambule est décelé, le noeud reste éveillé afin de recevoir les données qui suivront. Le préambule sert ainsi à synchroniser un ensemble de noeuds afin de s'assurer qu'ils soient prêts à recevoir les données envoyées par l'émetteur du préambule. La longueur du préambule doit évidemment être supérieure à la période d'échantillonnage du médium.

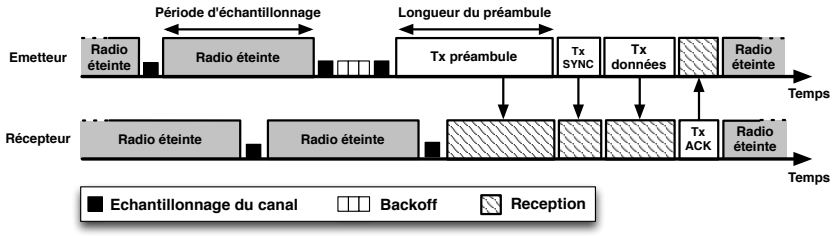


Figure 67: Le protocole B-MAC : les données sont envoyées après un préambule et un cours message de synchronisation (SYNC).

X-MAC [17] optimise B-MAC en utilisant un préambule stroboscopique (Figure 68) : le préambule est décomposé en de multiples paquets séparés par un intervalle. Le récepteur peut profiter de cet intervalle pour acquitter la réception de l'un de ces paquets. La longueur totale du préambule peut ainsi être réduite de manière significative, économisant de l'énergie aussi bien au niveau de l'émetteur que du récepteur.

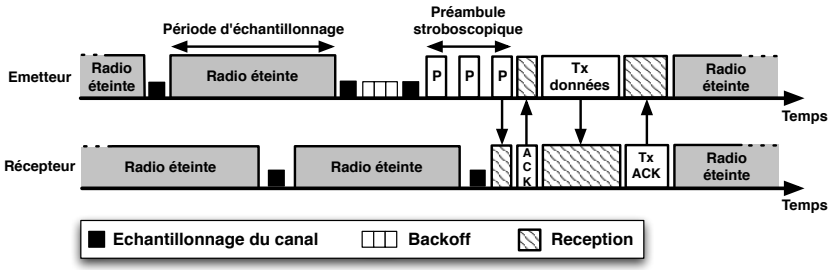


Figure 68: Le protocole X-MAC : le préambule stroboscopique permet de réduire l'écoute involontaire.

Les protocoles synchronisés s'organisent autour d'un horaire commun. Le temps est divisé en intervalles discrets, utilisés par les capteurs pour envoyer ou recevoir des données, ou pour éteindre leur radio. Les protocoles CSMA à intervalles tel que S-MAC [110] ainsi que les protocoles TDMA (e.g. TRAMA [80]) appartiennent à cette catégorie. Les protocoles à intervalles de temps rencontrent plusieurs difficultés lors d'une utilisation dans des réseaux dont la topologie ou le modèle de collection de données est dynamique. Par exemple, la synchronisation périodique indispensable à S-MAC peut empêcher un capteur mobile d'émettre ou de recevoir des données si celui-ci ne connaît pas la période utilisée dans le cluster où il se situe. De même, les protocoles TDMA s'accommodent difficilement de l'apparition de nouveaux capteurs ou d'une variation de trafic. En effet, ils nécessitent de recalculer et de redistribuer les intervalles de temps en fonction des besoins, afin de prendre en compte d'éventuels nouveaux participants dans les communications, ou de soustraire ceux n'étant plus à proximité. Les solutions proposées jusqu'à présent posent notamment le problème du passage à l'échelle : le nombre d'intervalles de la fenêtre d'émission étant borné, ce type de proposition reste difficilement extensible à de nombreux noeuds. De plus, le déploiement à grande échelle de ces protocoles nécessite des mécanismes de synchronisation temporelle, qui représentent encore aujourd'hui un défi majeur dans les réseaux sans fil multi-sauts.

Les protocoles par échantillonnage ne souffrent pas de ces limitations, et sont donc moins complexes à mettre en oeuvre à grande échelle. Dans nos travaux, nous nous intéresserons plus particulièrement à cette catégorie de protocoles. Nous avons notamment étudié leurs performances dans le cadre de scénarios dynamiques.

B.3 EVALUATION DES PROTOCOLES PAR ÉCHANTILLONNAGE

B.3.1 *L'accès au médium en environnement mobile*

À l'aide du simulateur de réseaux de capteurs sans fil WSNet [21], nous avons soulevé plusieurs problèmes peu considérés jusqu'à présent dans la littérature. Nous avons notamment évalué les pertes de paquets au niveau du capteur mobile lorsque celui évolue dans un réseau dont la densité de noeuds fixes et leur période d'envoi des données augmentent au fil des simulations. La Figure 69 détaille par exemple les pertes subies par noeud mobile opérant le protocole B-MAC (avec un préambule de 100 ms) en fonction de la taille moyenne de son voisinage. Le noeud mobile envoie toujours ses données avec une période égale à 1 s.

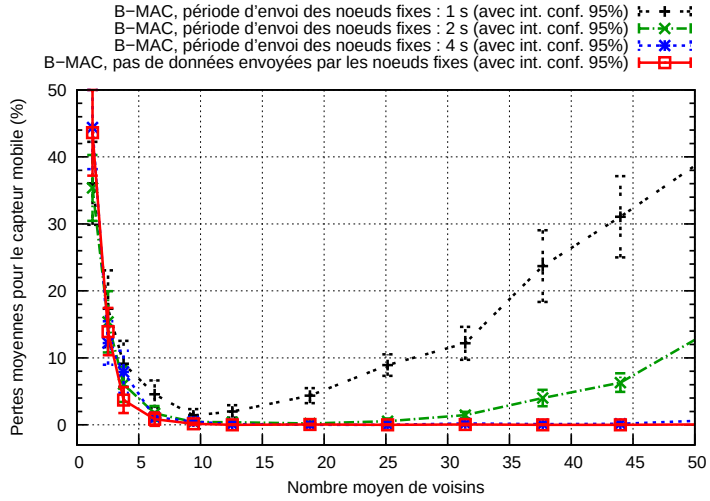


Figure 69: Pertes de paquets d'un capteur mobile utilisant B-MAC, en fonction du nombre moyen de voisins.

Nous remarquons notamment un taux de perte supérieur à 10% lorsque le nombre moyen de voisins du capteur mobile est inférieur à 3.77, ainsi que lorsque celui-ci est supérieur à 30 et que les communications dans le réseau sont fréquentes. Dans le premier cas, il s'avère que la densité du réseau n'est pas suffisante pour que le mobile soit toujours dans le voisinage d'autres capteurs lorsqu'il envoie ses données : les paquets sont naturellement perdus. Dans le deuxième cas, la majeure partie des paquets non-reçus en fin de simulation sont en fait encore dans la file d'attente de B-MAC. Le niveau élevé de compétition dans le réseau provoque une augmentation du délai moyen d'accès au médium pour le noeud mobile, qui peut dans certains cas dépasser sa période d'envoi des données. Cela explique le fait que la file d'attente du mobile se remplisse plus vite qu'elle ne se vide. Ainsi, en fonction de la taille de la file d'attente, les paquets peuvent soit être jetés, soit fortement retardés. Dû à la mobilité

du noeud, un envoi différé peut également provoquer un perte si la destination (choisie par la couche de routage) n'est plus valide au moment effectif de la transmission. Ceci est d'autant plus problématique selon l'importance des données du mobile. Afin de résoudre ces problèmes, nous proposons en Section B.4.1 une nouvelle méthode d'accès au médium destinée aux capteurs mobiles.

B.3.2 *Performances globales dans un scénario dynamique*

Nous avons démontré avec WSNet que les performances énergétiques des protocoles à préambules ne sont pas optimales lorsque des transmissions par rafales apparaissent dans le réseau. Pour cela, nous avons testé les protocoles versatiles B-MAC et X-MAC pré-configurés avec différentes longueurs de préambules, qui ont révélés qu'aucune configuration n'est réellement adéquate.

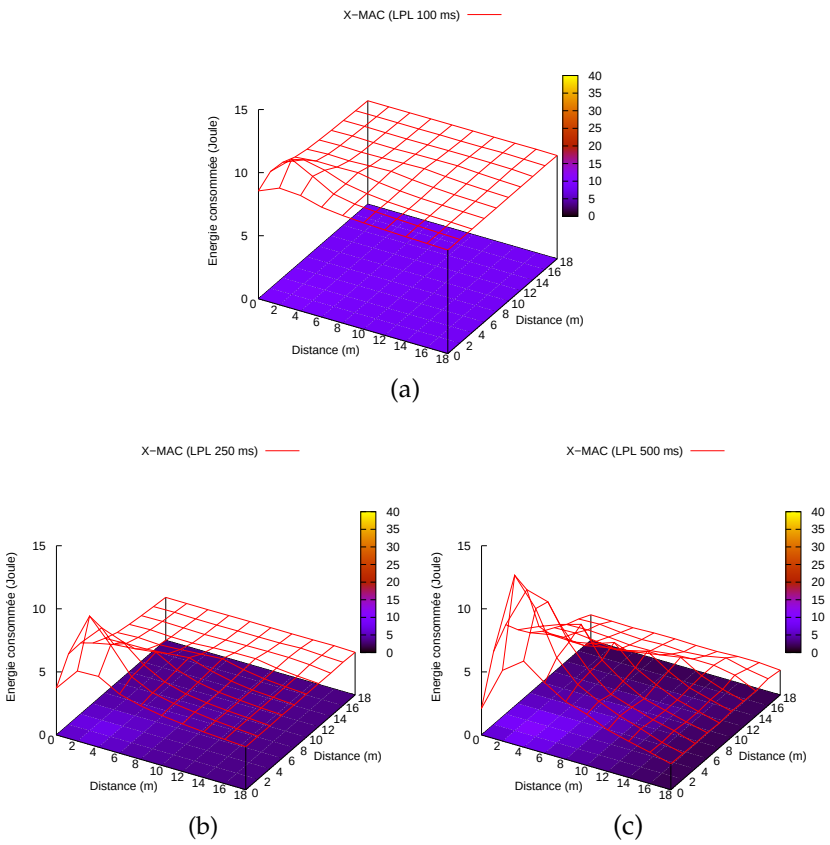


Figure 70: Carte de la consommation énergétique d'une grille de capteurs opérant X-MAC pour trois valeurs de LPL : (a) 100 ms, (b) 250 ms, (c) 500 ms.

Par exemple, la Figure 70 montre la consommation énergétique d'une grille de 10x10 capteurs opérant le protocole X-MAC, lorsque des évènements aléatoires surviennent dans le réseau.

Chaque évènement provoque une transmission en rafale à partir du capteur sur lequel l'évènement survient, jusqu'au puits situé dans le coins inférieur gauche de la grille. Trois valeurs de LPL (i.e. longueur du préambule et période d'échantillonnage) ont été testées : 100 ms, 250 ms, et 500 ms.

Lorsque de courts préambules sont utilisés, beaucoup d'énergie est gaspillée pour échantillonner le canal (les capteurs devant écouter le médium plus régulièrement). Cela réduit toutefois les temps de transmission et de réception sur les capteurs situés sur le chemin de routage des données. L'inverse se produit lorsque de long préambules sont utilisés. Les performances en termes de délais et de disponibilité du médium diminuent également lorsque la taille du préambule augmente. Les avantages des deux configurations (préambule court ou long) ne peuvent être associées en même temps sans modifier la configuration des capteurs pendant que le réseau opère. Un tel changement peut néanmoins empêcher les noeuds de communiquer ensemble, si par exemple l'émetteur utilise un préambule plus court que la période d'échantillonnage du récepteur. C'est pourquoi nous proposons en Section B.4.2 un algorithme qui permet d'adapter dynamiquement la taille du préambule tout en assurant la connectivité entre les noeuds.

B.4 CONTRIBUTIONS

B.4.1 *Machiavel et X-Machiavel*

Le rôle d'un capteur mobile peut s'avérer crucial dans certains scénarios, tels que la détection d'intrusion ou le suivi de cible. Il peut ainsi être nécessaire d'allouer une plus grande importance aux données qu'il transmet. C'est dans cette optique que nous avons élaboré le protocole *Machiavel*, une nouvelle méthode d'accès au médium pour capteurs mobiles. Comme détaillé sur la Figure 71, notre protocole autorise les noeuds mobiles à prendre possession du médium initialement réservé par un noeud fixe,

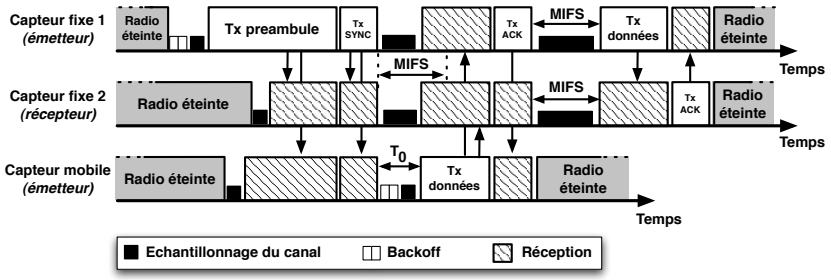


Figure 71: Le protocole Machiavel : le capteur mobile profite du court délai (MIFS) entre le message SYNC et les données du capteur fixe pour envoyer ses propres données.

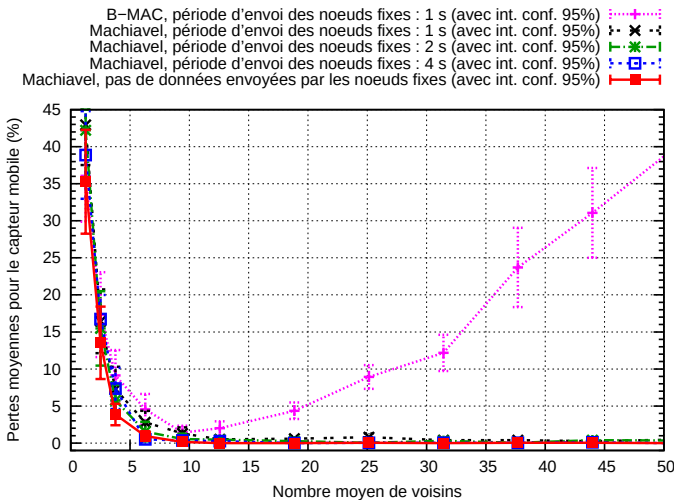


Figure 72: Pertes de paquets d'un capteur mobile utilisant Machiavel, en fonction du nombre moyen de voisins.

en envoyant leurs données entre le message SYNC et les données de ce noeud. Pour ce faire, les capteurs fixes respectent toujours un court délai (MIFS) après l'émission du SYNC.

À la différence d'un protocole par échantillonnage classique, Machiavel permet au noeud mobile d'émettre en étant assuré que ses voisins ont été correctement synchronisés tout en réduisant les délais d'accès au médium. Une évaluation du protocole nous a permis de démontrer ses bénéfices, notamment en réseaux denses. Par exemple, la Figure 72 démontre que les pertes du mobile sont significativement réduites, même lorsque la fréquence de collecte des données est élevée dans le réseau.

Une évolution de notre protocole, nommée *X-Machiavel* et basée sur X-MAC, nous a permis d'étendre le champ d'action de notre contribution à des topologies où le nombre de capteurs mobiles dépasse celui des noeuds fixes. X-Machiavel se base sur une infrastructure de capteurs fixes qui s'occupe du routage des données entre les noeuds. Sur un canal libre, les noeuds fixes peuvent réclamer les données des mobiles et les transférer vers leurs destinations respectives. Sur un canal occupé, les capteurs mobiles peuvent prendre possession du médium afin de s'intégrer dans l'ordonnancement des communications. X-Machiavel permet également aux noeuds mobiles de s'affranchir d'un protocole de routage tout en assurant la bonne réception de leurs données à 1 saut.

Nous avons comparé notre contribution avec le protocole X-MAC et démontré ses capacités à supporter un ratio de noeuds mobile plus importants. Par exemple, la Figure 73 montre que les pertes au niveau MAC peuvent être divisées par deux par rapport à X-MAC lorsque ce ratio est égal à 4. Etant donné que notre protocole affranchit le capteur mobile de l'utilisation d'un

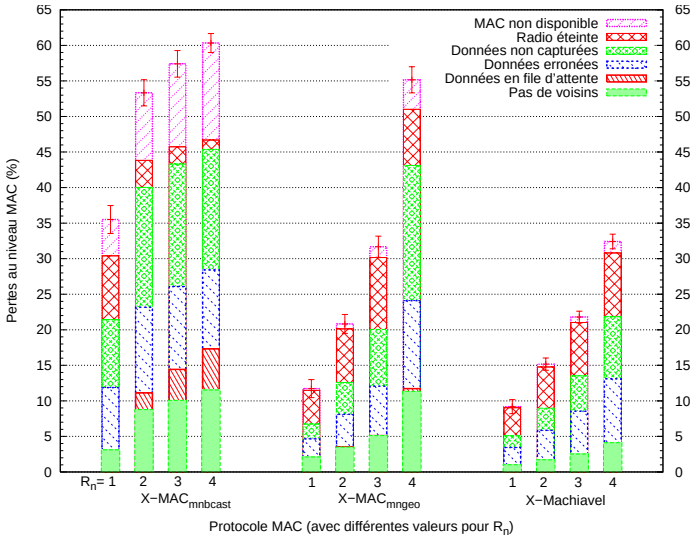


Figure 73: Pertes de paquets au niveau MAC pour un capteur mobile utilisant X-Machiavel, en fonction du ratio R_n de noeuds mobiles par rapport aux noeuds fixes.

protocole de routage, nous l'avons comparé à X-MAC$_{mngeo}$ (qui utilise un protocole de routage géographique sur les capteurs mobiles) et X-MAC$_{mnbcast}$ (qui diffuse les données en broadcast). Nous avons également constaté une réduction de 33% dans la consommation énergétique globale par rapport à X-MAC.

B.4.2 *BOX-MAC*

Les protocoles versatiles n'offrent pas de solutions satisfaisantes lorsque le modèle de collection de données évolue au cours du temps dans le réseau. Afin de rendre ces protocoles auto-adaptatifs, nous proposons un algorithme qui permet d'ajuster dynamiquement la période d'échantillonnage ainsi que la taille du préambule utilisés par les couches MAC des capteurs, tout en assurant la connectivité dans le réseau (Figure 74).

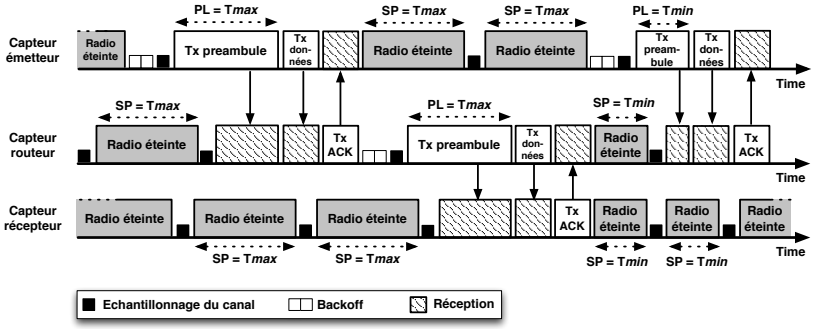


Figure 74: Auto-adaptation de la longueur du préambule (PL) et de la période d'échantillonnage du canal (SP).

Au départ, tous les capteurs du réseau opèrent avec une période d'échantillonnage longue (SP = T_{max}). Lors d'une transmission en rafale, l'émetteur envoie la première donnée en utilisant un long préambule (PL = T_{max}, afin de s'assurer que le récepteur le reçoive). Cette première émission exprime le désir implicite de passer à une taille de préambule plus courte pour les transmissions suivantes (PL = T_{min}). Le récepteur adapte donc sa période d'échantillonnage en conséquence (SP = T_{min}). En fonction du rôle du capteur (émetteur, récepteur final ou routeur), le capteur adapte respectivement PL, SP, ou les deux. L'émetteur et le récepteur passent à nouveau à une période d'échantillonnage et un préambule longs lorsque qu'aucune donnée n'a été envoyée ni reçue pendant un certain temps. En adaptant automatiquement ces paramètres, une configuration optimale peut être utilisée aussi bien sur le chemin de routage du réseau (grâce à un préambule court) qu'aux endroits moins actifs (en utilisant une période d'échantillonnage longue). L'efficacité énergétique ainsi que la connectivité entre les capteurs sont assurées grâce à une négociation réalisée de proche en proche le long du chemin de routage.

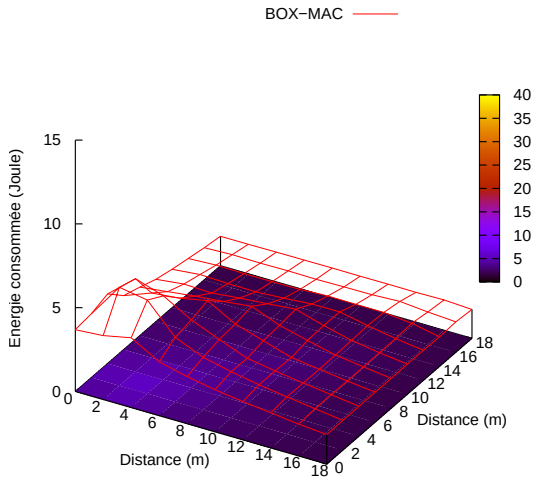


Figure 75: Carte de la consommation énergétique d'une grille de capteurs opérant BOX-MAC.

Les résultats des simulations effectuées avec WSNet montrent que notre contribution est particulièrement efficace lorsque des données sont envoyées en rafales lors d'évènements aléatoires dans le réseau. BOX-MAC (Burst-Oriented X-MAC, notre algorithme couplé au protocole X-MAC) a prouvé son efficacité énergétique tout en conservant un niveau de fiabilité comparable au protocole sur lequel il est appliqué. Par exemple, la Figure 75 montre la consommation énergétique globale d'un réseau opérant le protocole BOX-MAC dans les mêmes conditions que celles décrites dans la Section B.3.2. Nous pouvons affirmer que BOX-MAC est plus économe en énergie que n'importe laquelle des configurations détaillées précédemment dans la Figure 70.

B.5 CONCLUSION

Dans un avenir proche, les réseaux de capteurs seront sans doute déployés à grande échelle, mettant en scène des capteurs mobiles évoluant pendant plusieurs mois sans interventions extérieures. Nous nous sommes interrogés sur la capacité des protocoles de communication actuels à pouvoir intégrer les noeuds mobiles dans leurs algorithmes d'accès au médium, ainsi que sur l'impact de tels réseaux dynamiques au niveau des performances globales.

Utilisés dans ce cadre, les protocoles par échantillonnage ont révélé plusieurs problèmes de performance. Nos contributions Machiavel et X-Machiavel permettent notamment de mieux intégrer les capteurs mobiles au sein d'un réseau de noeuds fixes, en réduisant leurs pertes de paquets et leur délai d'accès au médium. Notre protocole auto-adaptatif BOX-MAC améliore les performances énergétiques globales de tels réseaux, notamment lorsque la charge fluctuante demande des méthodes d'accès différentes. Nous avons principalement travaillé avec les protocoles B-MAC et X-MAC, mais nous pensons que les principes derrière nos contributions pourraient aussi s'adapter à d'autres protocoles par échantillonnage.

Ces travaux pourraient bénéficier de plusieurs optimisations, notamment en utilisant des informations de niveau 3 afin d'affiner les algorithmes d'accès au médium. Nous souhaiterions évaluer l'ensemble de nos contributions sur la plateforme de capteurs à grande échelle Senslab [87], qui dispose de 1024 capteurs fixes et mobiles.

LIST OF ACRONYMS

API Application Programming Interface

ABNF Augmented Backus-Naur Form

BAN Body Area Network

BID Binding Identifier

BMA Bit-Map Assisted

B-MAC Berkeley MAC

BOB-MAC Burst-Oriented B-MAC

BOX-MAC Burst-Oriented X-MAC

BPSK Binary Phase-Shift Keying

CA Collision Avoidance

CALM Communication Access for Land Mobiles

CCA Clear Channel Assessment

CDMA Code Division Multiple Access

CN Correspondent Node

COA Care-of Address

CPU Central Processing Unit

CSMA Carrier Sense Multiple Access

CSMA-MPS CSMA with Minimum Preamble Sampling

CTS Clear To Send

DCF Distributed Coordination Function

DSMAC Dynamic S-MAC

DTN Delay-Tolerant Network

EA-ALPL Energy-Aware Adaptive LPL

EE Energy-Efficient

EMACS Eyes MAC

FDMA Frequency Division Multiple Access

FLAMA FLow-Aware Medium Access

FLEXIMAC Flexible MAC

GPS Global Positioning System

HA Home Agent

HOA Home Address

IPV6 Internet Protocol version 6

ITS Intelligent Transportation System

LLC Logical Link Control

LPL Low Power Listening

LQI Link Quality Indicator

MAC Medium Access Control

MAN MNN Add Notify

MCOA Multiple Care-of Addresses

MCU MicroController Unit

MEMAC Mobility Aware and Energy Efficient MAC

MFP-MAC Micro Frame Preamble MAC

MH-MAC1 Multimod-Hybrid MAC

MH-MAC2 Mobility Adaptive Hybrid MAC

MIFS Machiavel Inter-Frame Space

MIPV6 Mobile IPv6

MMAC Mobility-adaptive MAC

MMSN Multi-Frequency MAC for WSNs

MNN Mobile Network Node

MNP Mobile Network Prefix

MORS Mobile Router Status

MR Mobile Router

MS-MAC Mobility-aware S-MAC

NA Neighbor Advertisement

NEMO BS Network Mobility Basic Support

NIMH Nickel Metal Hydride

NRA Neighbor Router Advertisement

NS Neighbor Solicitation

OSI Open System Interconnection

PACT Power-Aware Cluster TDMA

PDA Personal Digital Assistant

PL Preamble Length

PMAC Pattern MAC

PSM Power Saving Mode

RAM Random Access Memory

RISC Reduced Instruction Set Computer

RTS Request To Send

SCP Scheduled Channel Polling

S-MAC Sensor MAC

SMACS Self-Organizing MAC for Sensor Networks

SP Sampling Period

STEM Sparse Topology and Energy Management

SYNC Synchronization

SYNCWUF Synchronized Wake-Up-Frame

TDMA Time Division Multiple Access

TICER Transmitted Initiated Cycled Receiver

T-MAC Timeout MAC

TRAMA Traffic-adaptive MAC

TSMP Time Synchronized Mesh Protocol

U-MAC Utilization-based MAC

WISEMAC Wireless Sensor MAC

WSN Wireless Sensor Network

WUF Wake-Up Frame

Z-MAC Zebra MAC

LIST OF FIGURES AND TABLES

LIST OF FIGURES

LIST OF TABLES

LIST OF PUBLICATIONS

INTERNATIONAL JOURNALS

An improved network mobility service for intelligent transportation systems. Romain Kuntz, Julien Montavont, Guillaume Schreiner, Thomas Noël, and David Binet. *Wiley InterScience Wireless Communications and Mobile Computing*, Special issue on Emerging Techniques for Wireless Vehicular Communications, November 2009.

Medium access control facing the reality of WSN deployments. Romain Kuntz, Antoine Gallais, and Thomas Noël. *ACM SIGCOMM Computer Communication Review*, 39(3):22–27, July 2009.

INTERNATIONAL CONFERENCES

CASINO: Creating Alea with a Sensor-based Interactive Network (Demo abstract). Julien Beaudaux, Antoine Gallais, Romain Kuntz, Julien Montavont, Thomas Noël, Damien Roth, Fabrice Theoleyre and Erkan Valentin. In *Sensys '10: Conference on Embedded Networked Sensor Systems*. ACM, November 2010.

Machiavel: Accessing the medium in mobile and dense WSN. Romain Kuntz and Thomas Noël. In *PIMRC '09: International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5. IEEE, September 2009.

Multiple mobile routers in NEMO: How neighbor discovery can assist default router selection. Romain Kuntz, Julien Montavont, and Thomas Noël. In *PIMRC '08: International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–6. IEEE, September 2008.

Versatile IPv6 mobility deployment with dual stack mobile IPv6. Romain Kuntz and Jean Lorchat. In *MobiArch '08: 3rd international workshop on Mobility in the evolving internet architecture*, pages 49–54. ACM, August 2008.

NATIONAL CONFERENCES

Machiavel, un protocole d'accès au médium orienté mobilité et réseaux de capteurs. Romain Kuntz and Thomas Noël. In *CFIP'09: Colloque francophone sur I'ingéniérie des protocoles*, pages 75–86, October 2009.

IETF INTERNET DRAFTS

Flow distribution rule language for multi-access nodes. Conny Larsson, Michael Eriksson, Koshiro Mitsuya, Kazuyuki Tasaka, and Romain Kuntz. *IETF Internet-Draft, February 2009*.

Mobile IPv6 flow routing over multiple care-of addresses. Michael Eriksson, Conny Larsson, and Romain Kuntz. *IETF Internet-Draft, June 2008*.

BIBLIOGRAPHY

[1] N. Abramson. The ALOHA system: another alternative for computer communications. In *AFIPS '70 fall joint computer conference*, pages 281–285. ACM, November 1970.

[2] G.-S. Ahn, S. G. Hong, E. Miluzzo, A. T. Campbell, and F. Cuomo. Funneling-MAC: a localized, sink-oriented MAC for boosting fidelity in sensor networks. In *SenSys '06: 4th International Conference on Embedded Networked Sensor Systems*, pages 293–306. ACM, October 2006.

[3] K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Elsevier Ad Hoc Networks*, 3(3):325–349, May 2005.

[4] M. Ali, U. Saif, A. Dunkels, T. Voigt, K. Römer, K. Langendoen, J. Polastre, and Z. A. Uzmi. Medium access control issues in sensor networks. *ACM SIGCOMM Computer Communication Review*, 36(2):33–36, April 2006.

[5] M. Ali and Z. A. Uzmi. Medium access control with mobility adaptive mechanisms for wireless sensor networks. *International Journal of Sensor Networks*, 1(3/4):134–142, 2006.

[6] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, J. Huang, R. Han, D. Lawrence, and K. Mohseni. Sensorflock: an airborne wireless sensor network of micro-air vehicles. In *SenSys '07: 5th International Conference on Embedded Networked Sensor Systems*, pages 117–129. ACM, November 2007.

[7] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella. Energy conservation in wireless sensor networks: A survey. *Elsevier Ad Hoc Networks*, 7(3):537–568, May 2009.

[8] X. aol ei Shi and G. Stromberg. SyncWUF: An ultra low-power MAC protocol for wireless sensor networks. *IEEE Transactions on Mobile Computing*, 6(1):115–125, January 2007.

[9] K. A. Arisha, M. A. Youssef, and M. F. Younis. Energy-aware TDMA-based MAC for sensor networks. In *IMPACCT '02: Workshop on Integrated Management of Power Aware Communications, Computing and Networking*. IEEE, May 2002.

[10] Atmel corporation. ATmega low-power microcontroller. http://www.atmel.com.

[11] A. Bachir, D. Barthel, M. Heusse, and A. Duda. Micro-frame preamble MAC for multihop wireless sensor networks. In *ICC '06: International Conference on Communications*, pages 3365–3370. IEEE, June 2006.

[12] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung. MAC essentials for wireless sensor networks. *IEEE Communications Surveys Tutorials*, 12(2):222–248, 2010.

[13] C. Baker, K. Armijo, S. Belka, M. Benhabib, V. Bhargava, N. Burkhart, A. Der Minassians, G. Dervisoglu, L. Gutnik, M. Haick, C. Ho, M. Koplow, J. Mangold, S. Robinson, M. Rosa, M. Schwartz, C. Sims, H. Stoffregen, A. Waterbury, E. Leland, T. Pering, and P. Wright. Wireless sensor networks for home health care. In *AINAW '07: 21st International Conference on Advanced Information Networking and Applications Workshops*, volume 2, pages 832 –837, 21-23 2007.

[14] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange. SensorScope: Out-of-the-box environmental monitoring. In *IPSN '08: 7th International Conference on Information Processing in Sensor Networks*, pages 332–343. ACM/IEEE, April 2008.

[15] L. Bernardo, R. Oliveira, M. Pereira, M. Macedo, and P. Pinto. A wireless sensor MAC protocol for bursty data traffic. In *PIMRC '07: International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5. IEEE, September 2007.

[16] P. Biondi and al. Scapy, the interactive packet manipulation program. http://www.secdev.org/projects/scapy/.

[17] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *SenSys '06: 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320. ACM, October 2006.

[18] R. Cardell-Oliver, K. Smettem, M. Kranz, and K. Mayer. Field testing a wireless sensor network for reactive environmental monitoring. In *ISSNIP '04: Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 7–12. IEEE, December 2004.

[19] M. Ceriotti, L. Mottola, G. Picco, A. Murphy, S. Guna, M. Corrà, M. Pozzi, D. Zonta, and P. Zanon. Monitoring

heritage buildings with wireless sensor networks: The Torre Aquila deployment. In *IPSN '09: 8th International Conference on Information Processing in Sensor Networks*. ACM/IEEE, April 2009.

[20] C. K. Chan, H. Peng, G. Liu, K. McIlwrath, X. F. Zhang, R. A. Huggins, and Y. Cui. High-performance lithium battery anodes using silicon nanowires. *Nature Nanotechnology*, 3:31–35, January 2008.

[21] G. Chelius and al. WSNet, an event-driven simulator for large scale WSNs. http://wsnet.gforge.inria.fr.

[22] S. Cho, J. NA, and C. Kim. A dynamic load sharing mechanism in multihomed mobile networks. In *ICC '05: International Conference on Communications*, volume 3, pages 1459–1463. IEEE, May 2005.

[23] N. Choi, J. Ryu, E. Paik, T. Kwon, and Y. Choi. A transparent failover mechanism for a mobile network with multiple mobile routers. *IEEE Communications Letters*, 11(7):604–606, July 2007.

[24] Crossbow Technology Inc. The TelosB wireless sensor board. http://www.xbow.com.

[25] D. Culler and al. TinyOS, the operating system for wireless embedded sensor networks. http://www.tinyos.net.

[26] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme. Robomote: Enabling mobility in sensor networks. In *IPSN '05: 4th International Symposium on Information Processing in Sensor Networks*, pages 404–409. ACM, April 2005.

[27] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. Network Mobility (NEMO) Basic Support protocol. IETF RFC 3963, January 2005.

[28] A. Dunkels and al. Contiki, the operating system for embedded smart objects. http://www.sics.se/contiki/.

[29] A. Dunkels, N. Finne, J. Eriksson, and T. Voigt. Run-time dynamic linking for reprogramming wireless sensor networks. In *SenSys '06: 4th International Conference on Embedded Networked Sensor Systems*, pages 15–28. ACM, October 2006.

[30] P. Dutta, D. Culler, and S. Shenker. Procrastination might lead to a longer and more useful life. In *HotNets-VI: Sixth Workshop on Hot Topics in Networks*. ACM, November 2007.

[31] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *IPSN '05: 4th International Symposium on Information Processing in Sensor Networks*, pages 497–502. ACM, April 2005.

[32] A. El-Hoiydi. ALOHA with preamble sampling for sporadic traffic in ad hoc wireless sensor networks. In *ICC '02: International Conference on Communications*, volume 5, pages 3418–3423. IEEE, April 2002.

[33] A. El-Hoiydi and J.-D. Decotignie. Low power downlink MAC protocols for infrastructure wireless sensor networks. *Mobile Networks and Applications*, 10(5):675–690, 2005.

[34] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The pothole patrol: using a mobile sensor network for road surface monitoring. In *MobiSys '08: International Conference on Mobile Systems, Applications, and Services*, pages 29–39. ACM, 2008.

[35] M. Eriksson, C. Larsson, and R. Kuntz. Mobile IPv6 flow routing over multiple care-of addresses. IETF Internet-Draft, June 2008.

[36] T. Ernst, K. Mitsuya, and K. Uehara. Network mobility from the InternetCAR perspective. *Journal of Interconnection Networks (JOIN)*, 4(3), September 2003.

[37] R. Flury and R. Wattenhofer. MLS: an efficient location service for mobile ad hoc networks. In *MobiHoc '06: 7th international symposium on Mobile ad hoc networking and computing*, pages 226–237. ACM, May 2006.

[38] K. Gill, S.-H. Yang, F. Yao, and X. Lu. A zigbee-based home automation system. *IEEE Transactions on Consumer Electronics*, 55(2):422–430, May 2009.

[39] G. P. Halkes and K. G. Langendoen. Crankshaft: An energy-efficient MAC protocol for dense wireless sensor networks. In *EWSN '07: 4th European Conference on Wireless Sensor Networks*. Blackwell, January 2007.

[40] Hamamatsu photonics. Optical technology and opto electronics. http://www.hamamatsu.com.

[41] C. Hartung, R. Han, C. Seielstad, and S. Holbrook. FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In *MobiSys '06: International Conference on Mobile Systems, Applications, and Services*, pages 28–41. ACM, June 2006.

[42] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys '04: International Conference on Mobile Systems, Applications, and Services*, pages 270–283. ACM, June 2004.

[43] K. Heurtefeux and F. Valois. Distributed localization protocol for routing in a noisy wireless sensor network. In *MSN '09: 5th International Conference on Mobile Ad-hoc and Sensor Networks*, pages 223–230. IEEE, 2009.

[44] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, November/December 2002.

[45] IEEE Computer Society. IEEE standard for information technology - telecommunications and information exchange between systems- local and metropolitan area networks-specific requirements part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, pages 1–305, September 2006.

[46] IEEE Computer Society. 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. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages 1–1184, June 2007.

[47] D. Johnson, C. Perkins, and J. Arkko. Mobility support in IPv6. IETF RFC 3775, June 2004.

[48] R. Jurdak, P. Baldi, and C. Videira Lopes. Adaptive low power listening for wireless sensor networks. *IEEE Transactions on Mobile Computing*, 6(8):988–1004, August 2007.

[49] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *IPSN '07: 6th International Conference on Information Processing in Sensor Networks*, pages 254–263. ACM, April 2007.

[50] P. R. Kumar and P. Varaiya. *Stochastic systems: estimation, identification and adaptive control*. Prentice-Hall Inc., 1986.

[51] R. Kuntz, A. Gallais, and T. Noël. Medium access control facing the reality of WSN deployments. *ACM SIGCOMM Computer Communication Review*, 39(3):22–27, July 2009.

[52] R. Kuntz, A. Gallais, and T. Noël. From versatility to auto-adaptation of the medium access control in wireless sensor networks. *Submitted to Elsevier Journal of Parallel and Distributed Systems*, Special Issue on Advances of Research in Wireless Access and Mobile Networks, 2010.

[53] R. Kuntz, J. Montavont, and T. Noël. Multiple mobile routers in NEMO: How neighbor discovery can assist default router selection. In *PIMRC '08: International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–6. IEEE, September 2008.

[54] R. Kuntz, J. Montavont, and T. Noël. Improving the medium access in highly mobile wireless sensor networks. In *Submitted to MASS '10: 7th International Conference on Mobile Ad-hoc and Sensor Systems*, San Francisco, CA, USA, November 2010. IEEE.

[55] R. Kuntz, J. Montavont, G. Schreiner, T. Noël, and D. Binet. An improved network mobility service for intelligent transportation systems. *Wiley InterScience Wireless Communications and Mobile Computing*, Special issue on Emerging Techniques for Wireless Vehicular Communications, November 2009.

[56] R. Kuntz and T. Noël. Machiavel: Accessing the medium in mobile and dense WSN. In *PIMRC '09: International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5. IEEE, September 2009.

[57] R. Kuntz and T. Noël. Machiavel, un protocole d'accès au médium orienté mobilité et réseaux de capteurs. In *CFIP'09: Colloque francophone sur I'ingéniérie des protocoles*, pages 75–86, October 2009.

[58] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *IPDPS '06: 20th International Parallel and Distributed Processing Symposium*, pages 8–15. IEEE, April 2006.

[59] C. Larsson, M. Eriksson, K. Mitsuya, K. Tasaka, and R. Kuntz. Flow distribution rule language for multi-access nodes. IETF Internet-Draft, February 2009.

[60] H. K. Le, D. Henriksson, and T. Abdelzaher. A practical multi-channel media access control protocol for wireless sensor networks. In *IPSN '08: 7th International Conference on Information Processing in Sensor Networks*, pages 70–81. IEEE, April 2008.

[61] W. Lee, A. Datta, and R. Cardell-Oliver. FlexiMAC: A flexible TDMA-based MAC protocol for fault-tolerant and energy-efficient wireless sensor networks. In *ICON '06. 14th International Conference on Networks*, volume 2, pages 1–6. IEEE, September 2006.

[62] J. Leguay, M. Lopez-Ramos, K. Jean-Marie, and V. Conan. An efficient service oriented architecture for heterogeneous and dynamic wireless sensor networks. In *SensApp '08: 3rd International Workshop on Practical Issues in Building Sensor Network Applications*. IEEE, October 2008.

[63] J. Li and G. Y. Lazarou. A bit-map-assisted energy-efficient MAC scheme for wireless sensor networks. In *IPSN '04: 3rd International Symposium on Information Processing in Sensor Networks*, pages 55–60. ACM, April 2004.

[64] E.-Y. Lin, J. Rabaey, and A. Wolisz. Power-efficient rendezvous schemes for dense wireless sensor networks. In *ICC '04: International Conference on Communications*, volume 7, pages 3769–3776. IEEE, June 2004.

[65] P. Lin, C. Qiao, and X. Wang. Medium access control with a dynamic duty cycle for sensor networks. In *WCNC '04: Wireless Communications and Networking Conference*, volume 3, pages 1534–1539. IEEE, March 2004.

[66] S. Mahlknecht and M. Böck. CSMA-MPS: A minimum preamble sampling MAC protocol for low power wireless sensor networks. In *International Workshop on Factory Communication Systems*, pages 73–80. IEEE, September 2004.

[67] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *WSNA '02: 1st International Workshop on Wireless Sensor Networks and Applications*, pages 88–97. ACM, September 2002.

[68] C. Merlin and W. Heinzelman. Duty cycle control for low-power-listening MAC protocols. In *MASS '08: 5th International Conference on Mobile Ad-hoc and Sensor Systems*, pages 497–502. IEEE, September 2008.

[69] C. Merlin and W. Heinzelman. Schedule adaptation of low-power-listening protocols for wireless sensor networks. *IEEE Transactions on Mobile Computing*, 9(5):672 –685, May 2010.

[70] E. Meshkova, K. Rerkrai, J. Riihijärvi, and P. Mähönen. Optimizing component-oriented systems: A case study in wireless sensor networks. In *SIGCOMM '08, Demonstration Abstract*. ACM, August 2008.

[71] S. Nadas. Virtual router redundancy protocol version 3 for IPv4 and IPv6. IETF RFC 5798, March 2010.

[72] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor discovery for IP version 6 (IPv6). IETF RFC 4861, September 2007.

[73] C. Ng, T. Ernst, E. Paik, and M. Bagnulo. Analysis of multihoming in network mobility support. IETF RFC 4980, October 2007.

[74] G. Pei and C. Chien. Low power TDMA in large wireless sensor networks. In *MILCOM '01: Military Communications Conference*, volume 1, pages 347–351. IEEE, October 2001.

[75] H. Pham and S. Jha. Addressing mobility in wireless sensor media access protocol. In *ISSNIP '04: Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 113–118, December 2004.

[76] K. S. J. Pister and L. Doherty. TSMP: Time synchronized mesh protocol. In *DSN '08: International Symposium on Distributed Sensor Networks*, pages 391—398. IEEE/IFIP, November 2008.

[77] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: 2nd International Conference on Embedded Networked Sensor Systems*, pages 95–107. ACM, November 2004.

[78] A. Raja and X. Su. A mobility adaptive hybrid protocol for wireless sensor networks. In *CCNC '08: 5th Consumer Communications and Networking Conference*, pages 692–696. IEEE, January 2008.

[79] V. Rajendran, J. Garcia-Luna-Aveces, and K. Obraczka. Energy-efficient, application-aware medium access for sensor networks. In *MASS '05: 2nd International Conference on Mobile Ad-hoc and Sensor Systems*, pages 630–637. IEEE, November 2005.

[80] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. In *SenSys '03: 1st International Conference on Embedded Networked Sensor Systems*, pages 181–192. ACM, November 2003.

[81] S. Ray, D. Starobinski, and J. B. Carruthers. Performance of wireless networks with hidden nodes: a queuing-theoretic analysis. *Elsevier Computer Communications*, 28(10):1179–1192, 2005.

[82] I. Rhee, A. Warrier, M. Aia, and J. Min. Z-MAC: a hybrid MAC for wireless sensor networks. In *SenSys '05: 3rd International Conference on Embedded Networked Sensor Systems*, pages 90–101. ACM, November 2005.

[83] I. Rhee, A. Warrier, J. Min, and L. Xu. DRAND: distributed randomized TDMA scheduling for wireless ad-hoc networks. In *MobiHoc '06: 7th international symposium on Mobile ad hoc networking and computing*, pages 190–201. ACM, May 2006.

[84] K. Römer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, December 2004.

[85] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, 1(1):70–80, January 2002.

[86] Sensirion. Humidity, flow and pressure sensor solutions. http://www.sensirion.com.

[87] Senslab consortium. Senslab, the very large scale open wireless sensor network testbed. http://www.senslab.info.

[88] X. Shi, G. Stromberg, Y. Gsottberger, and T. Sturm. Wake-up-frame scheme for ultra low power wireless transceivers. In *Globecom '04: Global Telecommunications Conference*, volume 6, pages 3619–3623. IEEE, November 2004.

[89] V. Shnayder, B. rong Chen, K. Lorincz, T. R. F. Fulford-Jones, and M. Welsh. Sensor networks for medical care. In *Technical Report TR-08-05*. Division of Engineering and Applied Sciences, Harvard University, 2005.

[90] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7(5):16–27, October 2000.

[91] H. Soliman, G. Tsirtsis, N. Montavont, G. Giaretta, and K. Kuladinithi. Flow bindings in Mobile IPv6 and NEMO Basic Support. IETF Internet-Draft, March 2010.

[92] B. Sundararaman, U. Buy, and A. D. Kshemkalyani. Clock synchronization for wireless sensor networks: a survey. *Elsevier Ad Hoc Networks*, 3(3):281–323, May 2005.

[93] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *SenSys '04: 2nd International Conference on*

Embedded Networked Sensor Systems, pages 214–226. ACM, November 2004.

[94] Texas Instruments. Chipcon CC1100 low-power RF transceiver. http://www.ti.com/lit/gpn/cc1100.

[95] Texas Instruments. Chipcon CC2420 2.4GHz IEEE 802.15.4 RF transceiver. http://www.ti.com/lit/gpn/cc2420.

[96] Texas Instruments. MSP430 ultra-low-power microcontrollers. http://www.ti.com.

[97] S. Thomson, T. Narten, and T. Jinmei. IPv6 Stateless Address Autoconfiguration. IETF RFC 4862, September 2007.

[98] B. Thorstensen, T. Syversen, T.-A. Bjørnvold, and T. Walseth. Electronic shepherd – a low-cost, low-bandwidth, wireless network system. In *MobiSys '04: International Conference on Mobile Systems, Applications, and Services*, pages 245–255. ACM, June 2004.

[99] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *SenSys '05: 3rd International Conference on Embedded Networked Sensor Systems*, pages 51–63. ACM, November 2005.

[100] M. Tsukada, T. Ernst, R. Wakikawa, and K. Mitsuya. Dynamic management of multiple mobile routers. In *ICN '05: 13th International Conference on Networks*, volume 2, pages 1108–1113. IEEE, November 2005.

[101] USAGI Project, HUT and al. UMIP, the open-source Mobile IPv6 and NEMO Basic Support implementation for GNU/Linux. http://www.umip.org.

[102] T. van Dam and K. Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. In *SenSys '03: 1st International Conference on Embedded Networked Sensor Systems*, pages 171–180. ACM, November 2003.

[103] L. van Hoesel and P. Havinga. Poster abstract: A TDMA-based MAC protocol for WSNs. In *SenSys '04: 2nd International Conference on Embedded Networked Sensor Systems*. ACM, November 2004.

[104] VARTA Microbattery. Electrochemical energy storage systems. http://www.varta-microbattery.com.

[105] R. Wakikawa, V. Devarapalli, G. Tsirtsis, T. Ernst, and K. Nagami. Multiple care-of addresses registration. IETF RFC 5648, October 2009.

[106] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25, March/April 2006.

[107] H. Wu and Y. Pan. *Medium Access Control in Wireless Networks*. Nova Science Publishers, 2008.

[108] B. Yahya and J. Ben-Othman. An adaptive mobility aware and energy efficient MAC protocol for wireless sensor networks. In *ISCC '09: IEEE Symposium on Computers and Communications*, pages 15–21. IEEE, July 2009.

[109] S.-H. Yang, H.-W. Tseng, E.-K. Wu, and G.-H. Chen. Utilization based duty cycle tuning MAC protocol for wireless sensor networks. In *Globecom '05: Global Telecommunications Conference*, volume 6, pages 3258–3262. IEEE, November/December 2005.

[110] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, June 2004.

[111] W. Ye, F. Silva, and J. Heidemann. Ultra-low duty cycle MAC with scheduled channel polling. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 321–334. ACM, October 2006.

[112] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Elsevier Computer Networks*, 52(12):2292–2330, August 2008.

[113] K. Zen, D. Habibi, A. Rassau, and I. Ahmad. Performance evaluation of IEEE 802.15.4 for mobile sensor networks. In *WOCN '08. 5th International Conference on Wireless and Optical Communications Networks*, pages 1–5. IFIP, May 2008.

[114] M. Zhang, X. Du, and K. Nygard. Improving coverage performance in sensor networks by using mobile sensors. In *MILCOM '05: Military Communications Conference*, volume 5, pages 3335–3341. IEEE, October 2005.

[115] P. Zhang, C. M. Sadler, S. A. Lyon, and M. M. Tonosi. Hardware design experiences in zebranet. In *SenSys '04: 2nd International Conference on Embedded Networked Sensor Systems*, pages 227–238. ACM, November 2004.

[116] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *SenSys '03: 1st International Conference on Embedded Networked Sensor Systems*, pages 1–13. ACM, November 2003.

[117] T. Zheng, S. Radhakrishnan, and V. Sarangan. PMAC: an adaptive energy-efficient MAC protocol for wireless sensor networks. In *IPDPS '05: 19th International Parallel and Distributed Processing Symposium*, pages 1–8. IEEE, April 2005.

[118] 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 *INFOCOM '06: International Conference on Computer Communications*, pages 1–13. IEEE, April 2006.