Universidade do Minho
Escola de Engenharia

Nuno Alexandre Magalhães Pereira

Efficient Aggregate Computations in Large-Scale Dense Wireless Sensor Networks

April 2010

Universidade do Minho
Escola de Engenharia

Nuno Alexandre Magalhães Pereira

Efficient Aggregate Computations in Large-Scale Dense Wireless Sensor Networks

PhD Thesis

Thesis awarded under the MAP-i joint doctoral programme in Computer Science of Universidade do Minho, Universidade de Aveiro and Universidade do Porto (MAP).

Developed under the scientific supervision of:

Professor Eduardo Manuel de Médicis Tovar

Professor Paulo Manuel Martins de Carvalho

Research Centre in
Real-Time Computing Systems

This research was partially developed at the Real-Time Computing System Research Centre (CISTER), from the School of Engineering of the Polytechnic of Porto (ISEP/IPP)

Universidade do Minho
Escola de Engenharia

Nuno Alexandre Magalhães Pereira

Efficient Aggregate Computations in Large-Scale Dense Wireless Sensor Networks

Doctoral Committee:

Professor Adriano Jorge Cardoso Moreira, University of Minho, Portugal

Professor Eduardo Manuel de Médicis Tovar, CISTER/ISEP, Portugal

Professor Jorge Miguel Sá Silva, University of Coimbra, Portugal

Professor Manuel Alberto Pereira Ricardo, University of Porto, Portugal

Professor Paulo A. A. Pereira, University of Minho, Portugal

Professor Paulo Manuel Martins de Carvalho, University of Minho, Portugal

Professor Tarek F. Abdelzaher, University of Illinois, USA

This work was supported by the Fundação para a Ciência e Tecnologia under the grant BD/28987/2006.

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA TESE, APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE.

Universidade do Minho ____/____/_____

Assinatura: _____

Abstract

Assuming a world where we can be surrounded by hundreds or even thousands of inexpensive computing nodes densely deployed, each one with sensing and wireless communication capabilities, the problem of efficiently dealing with the enormous amount of information generated by those nodes emerges as a major challenge. The research in this dissertation addresses this challenge.

This research work proves that it is possible to obtain aggregate quantities with a time-complexity that is independent of the number of nodes, or grows very slowly as the number of nodes increases. This is achieved by co-designing the distributed algorithms for obtaining aggregate quantities and the underlying communication system. This work describes (i) the design and implementation of a prioritized medium access control (MAC) protocol which enforces strict priorities over wireless channels and (ii) the algorithms that allow exploiting this MAC protocol to obtain the minimum (MIN), maximum (MAX) and interpolation of sensor values with a time-complexity that is independent of the number of nodes deployed, whereas other state-of-the-art approaches have a time-complexity that is dependent on the number of nodes. These techniques also enable to efficiently obtain estimates of the number of nodes (COUNT) and the median of the sensor values (MEDIAN).

The novel approach proposed to efficiently obtain aggregate quantities in large-scale, dense wireless sensor networks (WSN) is based on the adaptation to wireless media of a MAC protocol, known as dominance/binary countdown, which existed previously only for wired media, and design algorithms that exploit this MAC protocol for efficient data aggregation. Designing and implementing such MAC protocol for wireless media is not trivial. For this reason, a substantial part of this work is focused on the development and implementation of WiDom (short for *Wireless Dominance*) - a wireless MAC protocol that enables efficient data aggregation in large-scale, dense WSN.

An implementation of WiDom is first proposed under the assumption of a fully connected network (a network with a single broadcast domain). This implementation can be exploited to efficiently obtain aggregated quantities. WiDom can also implement static priority scheduling over wireless media. Therefore, a schedulability analysis for WiDom is also proposed. WiDom is then extended to operate in sensor networks where a single transmission cannot reach all nodes, in a network with multiple broadcast domains.

These results are significant because often networks of nodes that take sensor readings are designed to be large scale, dense networks and it is exactly for such scenarios that the proposed distributed algorithms for obtaining aggregate quantities excel. The implementation and test of these distributed algorithms in a hardware platform developed shows that aggregate quantities in large-scale, dense wireless sensor systems can be obtained efficientlly.

Keywords: Medium Access Control (MAC), Data Processing, Wireless Sensor Networks, Cyber-Physical Systems, Data aggregation.

Resumo

É possível prever um mundo onde estaremos rodeados por centenas ou até mesmo milhares de pequenos nós computacionais densamente instalados. Cada um destes nós será de dimensões muito reduzidas e possui capacidades para obter dados directamente do ambiente através de sensores e transmitir informação via rádio. Frequentemente, este tipo de redes são denominadas de redes de sensores sem fio. Perante tal cenário, o problema de lidar com a considerável quantidade de informação gerada por todos estes nós emerge como um desafio de grande relevância. A investigação apresentada nesta dissertação atenta neste desafio.

Este trabalho de investigação prova que é possível obter quantidades agregadas com uma complexidade temporal que é independente do número de nós computacionais envolvidos, ou cresce muito lentamente quando o número de nós aumenta. Isto é conseguido através uma co-concepção dos algoritmos para obter quantidades agregadas e do sistema de comunicação subjacente. Este trabalho descreve (i) a concepção e implementação de um protocolo de acesso ao meio que garante prioridades estáticas em canais de comunicação sem fio e (ii) os algoritmos que permitem tirar partido deste protocolo de acesso ao meio para obter quantidades agregadas como o mínimo (MIN), máximo (MAX) e interpolação de valores obtidos a partir de sensores ambientais com uma complexidade que é independente do número de nós computacionais envolvidos. Estas técnicas também permitem obter, de forma eficiente, estimativas do número de nós (COUNT) e a mediana dos valores dos sensores (MEDIAN).

A abordagem inovadora, proposta para obter de forma eficiente quantidades agregadas em redes de sensores sem fio de larga escala, é baseada na adaptação para meios de comunicação sem fio de um protocolo de acesso ao meio anteriormente apenas existente em sistemas cablados, e na concepção de algoritmos que tiram partido deste protocolo para agregação de dados eficiente. A concepção e implementação de tal protocolo de acesso ao meio não é trivial. Por esta razão, uma parte substancial deste trabalho é focada no desenvolvimento e implementação de um protocolo de acesso ao meio que permite agregação de dados eficiente em redes de sensores sem fio densas e de larga escala. Esta implementação é denominada de WiDom.

A implementação do WiDom apresentada foi inicialmente desenvolvida assumindo que a rede é totalmente ligada (uma transmisão de um nó alcança todos os outros nós). Esta implementação pode ser explorada para obter quantidades agregadas de forma eficiente. Adicionalmente, o protocolo WiDom pode implementar escalonamento utilizando prioridades fixas, permitindo a proposta de uma análise de resposta temporal. Neste trabalho, o WiDom é também estendido para funcionar em redes onde a transmissão de um nó não pode alcançar todos os outros nós.

Os resultados apresentados neste trabalho são relevantes porque as redes de sensores sem fio são frequentemente concebidas para serem densas e de larga escala. É exactamente nestes casos que os algoritmos propostos para obter quantidades agregadas de forma eficiente apresentam maiores vantagens. A implementação e teste destes algoritmos distribuídos numa plataforma especialmente desenvolvida para o efeito demonstra que de facto podem ser obtidas quandidades agregadas de forma eficiente, mesmo em redes de sensores sem fio densas e de larga escala.

Palavras-Chave:Métodos de Acesso ao Meio, Processamento de Dados, Redes de Sensores sem Fio, Sistemas Cíber-Físicos, Agregação de Dados.

Acknowledgment

The research work leading to this thesis was largely developed at the Real-Time Computing Systems Research Centre (CISTER), from the School of Engineering of the Polytechnic Institute of Porto (ISEP/IPP). Officially, I had two exceptional supervisors: Prof. Eduardo Tovar, which lead my work at CISTER and Prof. Paulo Carvalho. Nevertheless, I would like to thank first and foremost my third, unofficial, supervisor: Björn Andersson. He was the precursor of this work and accompanied its development with enthusiasm, generosity and always showing, by his own example, that it takes a lot of hard work to develop research. I will venture borrowing some famous words to say that I have stood on the shoulders of a giant and that was what allowed me to develop this work (hopefully I saw a little bit further, but that judgment is left for others). Björn, my debt to you is enormous.

Prof. Eduardo Tovar provided me invaluable support. Not only by directly guiding my work, which is already a lot, but also by being the foremost responsible for the creation of the great environment that CISTER provides. Thank you for everything and special thanks for the effort put in reviewing my thesis in record time.

I also owe special thanks to Prof. Paulo Carvalho, for the discussions, reviewing work, support and patience for backing the bureaucratic difficulties.

Thanks to all the people at CISTER for all the great support at various levels. I would like to specially address a few of them. Thanks to Ricardo Gomes, the person behind the platform presented in Chapter 6, which he developed during his masters as a student under the guidance of Björn Andersson and myself. Thanks to Filipe Pacheco for his collaboration in part of the work. My recognition to Wilfried Elmenreich's contribution during his visit to CISTER. My sincere thanks to Mário Alves, for carefully following the development of this research. My thanks also to Sandra Almeida for the administrative support. To all others that were available to engage in fruitful discussions, teach me things, helped with lectures and any other assistance, I have not forgotten you! Thank you very much.

I was also lucky to meet some brilliant people during my stay in Pittsburg. My acknowledgement to Prof. Raj Rajkumar and the people in his group (the Real-Time and Multimedia Systems Lab from the Carnegie-Mellon University) for hosting me and providing a great environment for research and discussions. Special thanks here to Anthony Rowe, for his outstanding collaboration and for being a first-class host.

My gratitude also goes to Eduarda. We are not together anymore, but it would not be fair to forget all your support through the most crucial years in the development of this work.

Finally, my enormous gratitude to my Parents. For everything.

Contents

List of Figures

List of Tables

List of Acronyms

ALOHA	The ALOHA protocol is a medium access control protocol developed for the ALOHAnet, a pioneering computer networking system developed at the University of Hawaii.
ADC	Analog to Digital Converter
AM	Amplitude Modulation
AP	Access Point
API	Application Programming Interface
Atmega128	8-bit Microcontroller widely used in current wireless sensor network platforms
BS	Base Station
CAN	Controller Area Network, a bus network originally developed for the automotive industry that implements dominance/binary countdown for the medium access
CAP	Contention Access Period
CC2420	Radio transceiver widely used in current wireless sensor network platforms
CCA	Clear Channel Assessment
CDS	Connected Distributed Set
COTS	Commercial-off-the-shelf
COUGAR	An approach to in-network query processing in sensor networks.
COUNT	The number of nodes or proposed values (an aggregate quantity)
CPS	Cyber-Physical Systems
CPU	Central Processing Unit
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CTS	Clear-To-Send
DS	Dominating Set
EDF	Earliest Deadline First
FAMA-NCS	Floor Acquisition Multiple Access with Non-persistent Carrier Sensing
FTSP	Flooding Time Synchronization Protocol
GPS	Global Positioning System
GSM	Global System for Mobile communications
GTS	Guaranteed Time Slot
HPL	Hardware Presentation Layer

I-EDF	Implicit Earliest Deadline First, a MAC protocol that implements EDF scheduling
IEEE 802.11	Set of standards carrying out wireless local area network computer communication
IEEE 802.11e	Set of Quality of Service enhancements for wireless local area network applications through modifications to the media access control layer
IEEE 802.15.4	Set of standards which specifies the physical layer and media access control for low-rate wireless personal area networks
IFS	Inter-Frame Spacing
LEACH	Low Energy Adaptive Clustering Hierarchy
LR-PAN	Low-Rate wireless Personal Area Network
MAC	Medium Access Control
MACA	Multiple Access Control Avoidance
MACA-BI	Multiple Access Control Avoidance - By Invitation
MACAW	Multiple Access with Collision Avoidance for Wireless
MBD	Multiple Broadcast Domain
CDS	Connected Dominating Set
MCDS	Minimum Connected Dominating Set
MDS	Minimum Dominating Set
MVDS	Minimum Virtual Dominating Set
PAN	Personal Area Network
PEDAMACS	Power Efficient and Delay Aware Medium Access Control
PEGASIS	Power- Efficient Gathering in Sensor Information System
RBS	Reference Broadcast Synchronization
RSSI	Received Signal Strength Indicator
RTOS	Real-Time Operating System
RTS	Request-To-Send
SBD	Single Broadcast Domain
TDMA	Time Division Multiple Access
TPSN	Timing-sync Protocol for Sensor Networks
TRAMA	Tree-Search Resource Auction Multiple Access
WSN	Wireless Sensor Network

List of Symbols

α	An upper bound on the time-of-flight between two arbitrary nodes
C_i	Time to transmit a message from message stream i
C_i'	Time to perform the tournament when nodes are already synchronized and transmit a message
C_i''	Time to transmit a message and perform the tournament when nodes are not yet synchronized
CLK	Granularity of the clock
D_i	The relative deadline of a message from stream i in the system
E	Timeout to cope with synchronization imperfections (such as clock inaccuracies and transmit/receive switching times)
ε	Real-time clock error
F	Initial idle period
(x_i,y_i)	The x and y coordinates of a node i
$f(x,y)$	Denotes the function that interpolates the sensor data
G	Guarding time interval between bits
H	Duration of a priority bit
$hp(i)$	The set of all message streams with a higher priority than i
L	Time for the execution of the protocol
L_i	The length of the longest level-i busy period in non-preemptive context
$lp(i)$	The set of all message streams with a lower priority than message stream i
m	The (actual) number of nodes in the system
MAX	The maximum of the proposed values (sensor values or other; an aggregate quantity)
`MAX_TC`	Number of tournaments after which a transition that forces the protocol to try to reset its state, for error recovery purposes
MAXNNODES	An upper bound on the number of nodes m in the system
MAXP	The maximum value of the priorities; MAXP $= 2^{npriobits} - 1$
MAXS	Maximum sensor value on the platform MAXS $= 2^{NADCBITS} - 1$
MEDIAN	The median of the proposed values (sensor values or other; an aggregate quantity)
MIN	The minimum of the proposed values (sensor values or other; an aggregate quantity)

n	The number of message streams in the system
N_i	A node i in the system
NADCBITS	The number of bits of the ADC on the platform
npriobits	Number of priority bits
`prio[1..npriobits]`	An array of bits from 1 to *npriobits*, where the most significant bit is `prio[1]`. This array of bits holds the priority used in the tournament
Q	Defines the releases to be analysed in the schedulability analysis
Q_{bit}	The granularity of the time used in the schedulability analysis
$Q_{trnmt-SBD}$	Time to perform a tournament in WiDom-SBD considering the overhead of the protocol when nodes are not synchronized
$Q_{trnmt-MBD}$	Time to perform a tournament in WiDom-MBD considering the overhead of the protocol when nodes are not synchronized
R_{co}	The maximum range at which two nodes N_i and N_j can communicate reliably
R_{cs}	The maximum range at which N_i can detect a transmission from N_j
R_{it}	The maximum range between nodes N_j and N_k such that simultaneous transmissions to N_j will collide with N_k
R_i	An upper bound on the response time of a message stream i in the system
r_i	The maximum (observed) response time of a message stream i in the system
T_{CS}	Time to detect that a carrier wave is being transmitted
T_{RX}	Time to switch to reception mode
T_{RXTX}	Maximum between T_{RX} and T_{TX} ($\max\{T_{RX}, T_{TX}\}$)
T_{TX}	Time to switch to transmission mode
T_i	The minimum inter-arrival time of a message stream i in the system
`WINNER`	A boolean variable that holds the stated of the arbitration
$winner\_prio[i]$	Bit i in the integer holding the priority of the winner of the arbitration
$w_{i,q}$	Waiting time window in the schedulability analysis
x	A clock, used in the time automaton

Chapter 1

Introduction

Contents

1.1 Motivation and Objectives

Microprocessors are everywhere. Nowadays, we can find computing capabilities in everyday physical objects as diverse as mobile phones, digital personal assistants, gaming platforms, household appliances or cars, just to name a few examples.

Computing-enabled physical objects often have to deal with physical processes and tightly integrate computing with the physical world via sensors and actuators. The integration of physical processes and computing is not a new problem. Embedded systems, which have been in place long ago, often combine physical processes with computing. However, with the massive deployment of networked embedded computing devices, we are observing the next step in the evolution of embedded computing. The term Cyber-Physical Systems (CPS) has been used to describe these pervasive computing systems, where emphasis is put on the physical, real-time and embedded aspects [1].

Such large-scale, sensor-rich networked systems will generate an enormous amount of sensor data [2], and handling such amounts of data introduces significant challenges. One approach to deal with the amount of data generated in these systems is to perform in-network data aggregation. Instead of collecting data from all nodes to a central point, in-network data aggregation applies a data-reduction function to the data traveling through the network such that the total number of messages transmitted is reduced [3].

Despite the previous research developed in the field of data aggregation, its performance is limited by the fact that, nodes in the same radio broadcast range cannot transmit in parallel, hence the time-complexity still depends on the number of sensor nodes. If we envision scenarios where even a small area may contain several tens of sensor nodes, the advantages of typical data aggregation solutions found in the literature are significantly impaired.

It is in this context that the research work described addresses the problem of performing scalable and efficient aggregate quantities (e.g., the minimum, maximum

or median of the values proposed by all nodes) in dense networks. Here "efficient" means that the desired computation should be performed while consuming very little resources (such as energy, communication links, memory and processor) and "scalable" means that the consumption of resources should increase slowly or not at all as the number of sensor readings to be processed and/or the number of embedded computer nodes increases.

To illustrate this concept, consider a large-scale dense networked sensor system, whose nodes have a common sensing goal to measure temperature. Now consider the problem of computing a simple aggregate quantity: the minimum (MIN) sensed temperature among the nodes at some given moment. Computing MIN seems trivial, but for dense and large-scale systems, it poses an important problem: communicating sensor data individually makes the time-complexity of computing MIN a function of the number of nodes. This is true for any data aggregation mechanism employed (further details can be found in Section 2.2).

This research work aims at being able to validate and explore the following hypothesis:

> *Is it possible to efficiently obtain aggregate quantities with a time-complexity that is independent of the number of sensor nodes?*

In other words, and taking the example of MIN, we aim at computing MIN with a time-complexity that is equivalent to the time of transmitting a single message, even if tens or thousands of nodes share the same radio broadcast range.

Obtaining scalable and efficient aggregate quantities in large-scale dense networked sensor systems requires tight integration between the data aggregation techniques and communication mechanisms. This is a key observation underlying this research work, where the approach to obtain scalable and efficient aggregate quantities in large-scale dense networked sensor systems is co-designing (i) distributed algorithms to obtain aggregate quantities and (ii) the underlying communication services.

The main objective of this thesis is to demonstrate that is possible to obtain aggregate quantities efficiently by co-designing distributed algorithms for data aggregation with the underlying communication services. The approach to achieve this includes developing a prioritized MAC protocol and design distributed algorithms that exploit this MAC protocol. The next subsection presents more details on this approach.

1.2 Research Approach

This research work explores mechanisms for obtaining aggregate quantities that are efficient, even in very dense networks. The efficiency of traditional data aggregation mechanisms results from applying data reduction functions to data coming from different sources, and from exploiting the opportunities for parallel transmissions. In the extreme case where all nodes are in the same broadcast domain, nodes cannot transmit in parallel and there are no opportunities for traditional data aggregation techniques to apply a data reduction function.

The novel approach explored in this thesis is based on the adaptation to wireless media of a family of medium access control (MAC) protocols. This family of protocols is known as dominance or binary countdown protocols [4] and is already present in wired networking solutions: the Controller Area Network (CAN) technology [5]. We then design distributed algorithms that exploit the MAC protocol to efficiently obtain aggregate quantities.

Dominance/binary countdown protocols can be exploited to efficiently obtain a range of aggregate quantities. Let us briefly exemplify, to give further intuition, the case of MIN, which can be obtained with a time-complexity that is equivalent to the time of transmitting a single message. This is illustrated in Figure 1.1, where all nodes are in the same broadcast domain. Suppose that the temperature values are coded as n-bit integers. Starting with the most significant bit first, let each node send the temperature reading bit by bit. Consider also that, for each transmitted bit, nodes read the resulting value in the channel (something straightforward in a wired medium)

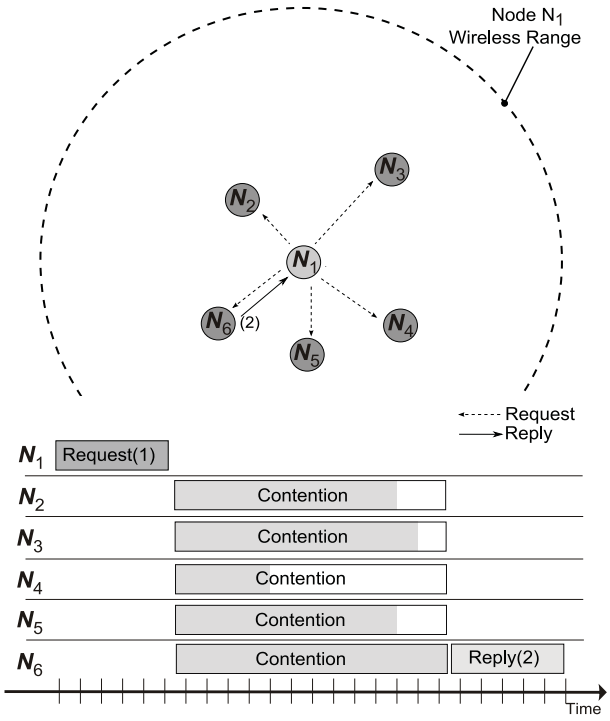


Figure 1.1: Data Aggregation by Exploiting a Prioritized MAC.

and the channel implements a logical AND of the transmitted bits. Furthermore, if a node reads '0' and is transmitting a '1', it stops transmitting. Then, at the end of the transmission of the n bits, the "observed" value in the channel will correspond to the MIN. It is as if all temperature readings were transmitted in parallel at the same time, and the resulting value of this non-destructive collision is a useful aggregate quantity.

It is based on this concept that the novel distributed algorithms proposed in Chapter 5 of this dissertation are designed upon. To accomplish this proposal, first it is necessary to design a MAC protocol that implements dominance/binary countdown in wireless environments, and then develop the algorithms to exploit that MAC protocol. However, designing and implementing such MAC protocol for wireless media is not trivial. For this reason, a substantial part of this work is focused on the development of such MAC protocol.

First, the problem is tackled assuming that all nodes belong to a *single broadcast domain* (SBD; treated in Chapter 3 and part of Chapter 5). Nodes are in a SBD

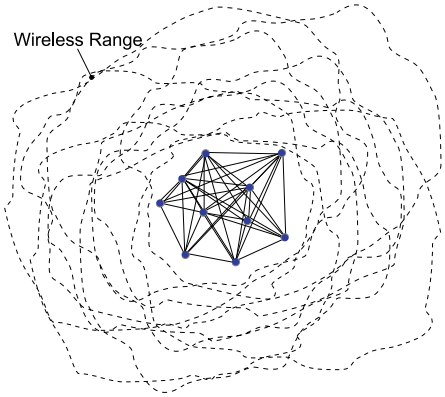


Figure 1.2: Illustration of a Network with a Single Broadcast Domain.

when (i) a wireless broadcast made by one node reaches all other nodes in the same broadcast domain and (ii) if a node transmits a packet, then it can be correctly received by another node in the same broadcast domain only if the transmission of the packet does not overlap in time with another packet transmission.

Figure 1.2 illustrates a SBD network. The radio ranges of the nodes are represented by the dashed lines. All nodes are inside the intersection of all radio ranges. The links between the nodes are also represented in Figure 1.2. There is a link between every pair of nodes. Contrarily to most figures in this dissertation, in Figure 1.2, radio ranges are illustrated with an irregular pattern. This is deliberately done to stress that, in this research work, there is no assumption about regular propagation patterns of the radio signals.

Achieving dominance in the wireless domain is challenging. To begin with, it is not possible to directly translate the behavior of wired protocols, as these require that nodes are able to transmit and receive at the same time. This is not possible in common radio transceivers, because the transmitted energy is much higher than the received energy. For this reason, dominance in wireless systems was achieved using a simple principle: when the transmitted bit is dominant, a pulse of a carrier wave is transmitted and there is no need to sense the medium. Conversely, when the bit to transmit is recessive, nothing has to be effectively sent, instead only the medium state has to be sensed.

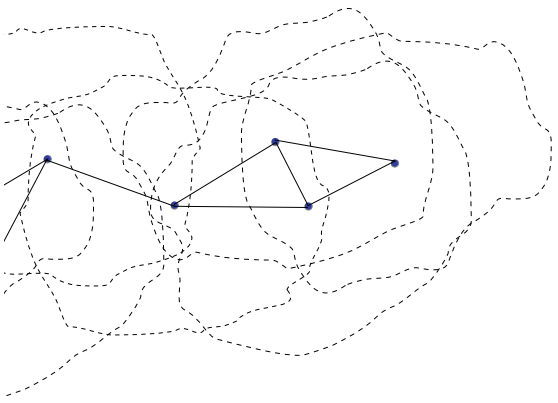


Figure 1.3: Illustration of a Network with Multiple Broadcast Domains.

Although the concept and approach sounds simple, a number of difficulties must be solved when proposing the design of a correct dominance protocol for wireless networks. These include achieving proper synchronization between the nodes, defining the parameters of the protocol such that clock inaccuracies, time-of-flight and other real-world effects are dealt with and how to perform reliable carrier detection. These aspects are addressed in Chapter 3 of this dissertation, where an implementation of dominance protocols in wireless media − WiDom (short for *wireless dominance*) − is presented.

This work also deals with the case of networks with *multiple broadcast domains* (MBD; addressed in Chapter 4 and part of Chapter 5). Considering MBD is important because it will be difficult to make the SBD assumption hold in a large number of networks deployed in the real-world. Nodes are in a MBD network if it holds that a wireless broadcast made by one node cannot reach all nodes in the network. Figure 1.3 illustrates an example of a MBD network. Such networks suffer from the well known hidden node problem (discussed later in Section 2.3.2). This is a challenge that needs to be solved when considering the extention of WiDom to MBD.

While a significant effort of this research work is put into designing novel distributed algorithms to obtain aggregate quantities in a SBD, local aggregation between nodes in geographic proximity can be used as an intermediate step to compute aggregated quantities among all nodes in a multihop network; hence the solution to the problem

of computing aggregated quantities in a SBD forms a relevant building block for large-scale data aggregation in multihop networks. This challenge is also tackled in this dissertation (in Chapter 5).

A final note on the MAC protocol that implements dominance/binary countdown in wireless media (WiDom). One important property of WiDom is that it allows enforcing static priorities. Therefore, it enables, for the first time, static priority scheduling over wireless media. This is also a relevant characteristic in emerging embedded systems because these systems deal with the physical world, therefore one important requirement to be met is that their data services are able to meet timing constraints [1]. The research approach also takes this property into account, and a response-time analysis for the proposed MAC protocol is also developed.

1.3 Contributions

This dissertation proves that it is possible to compute aggregate quantities with a time-complexity that is independent of the number of sensor nodes, or that the time-complexity grows very slowly as the number of nodes increases. This is achieved by closely articulating the distributed algorithms to obtain aggregate quantities and the underlying communication services. In this thesis we reason on the design and implementation of a prioritized MAC protocol which enforces strict priorities over wireless channels and on the design of algorithms that efficiently exploit this MAC protocol to obtain, in a SBD, the minimum (MIN), maximum (MAX) and interpolation of sensor values with a time-complexity that is independent of the number of sensor nodes (it depends only on the sensor value range). These techniques also enable to efficiently obtain estimates of the number of nodes (COUNT) and the MEDIAN. For MBD, the time-complexity of the proposed distributed algorithms developed also depends on the network diameter.

The research contributions are discussed in more detail in Chapter 7 and a full list of publications is included in Appendix A. Below we briefly summarize the main

research contributions.

Adaptation of Dominance Protocols to Wireless Media (publications [6, 7]). This research work introduces an adaptation of a dominance protocols for wireless media, which existed previously only for wired media. The implementation of a dominance protocol for wireless media was named WiDom and was initially proposed under the assumption of a SBD [6, 7]. WiDom can be exploited to efficiently obtain aggregated quantities (this is demonstrated in Chapter 5), and it is also useful to provide pre-runtime guarantees for sporadic messages streams. A schedulability analysis for WiDom was developed accordingly.

Extension of WiDom to Support Multiple Broadcast Domains (publication [8]). To cope with larger geographical areas, networks with *multiple broadcast domains* (MBD) need to be considered. An extension of WiDom for wireless networks with MBD was also proposed. The proposed solution is the first prioritized and collision-free MAC protocol designed to successfully deal with hidden nodes without relying on out-of-band signaling [8].

Improving the Reliability of WiDom in SBD (publication [9]). The techniques employed to solve the hidden node problem in [8] can also be adapted to improve the reliability of the protocol in a SBD. The proposed solution has a result that is similar to a cooperative relaying scheme, where several nodes can participate in the transmission of the priority bits [9].

Scalable and Efficient Aggregate Quantities (publications [10, 11]). By exploiting dominance protocols it is straightforward to demonstrate that, in a SBD, the minimum value (MIN) can be obtained with a time-complexity that is $O(npriobits)$, where *npriobits* is the number of bits used to represent the sensor data (the same technique can be applied to obtain the maximum value (MAX); see Chapter 5). techniques to efficiently compute more complex aggregate quantities such as the number of nodes (COUNT), MEDIAN and interpolation by exploiting dominance protocols were also implemented in the wireless domain [10]. Finally, the techniques employed to obtain aggregate quantities in a SBD were also extended for multihop networks [11].

The algorithms proposed for MBD have a time-complexity that only depends on the network diameter and on the value range of the sensor readings.

1.4 Outline

The remainder of this dissertation is organized as follows. Chapter 2 provides background material on two relevant areas for the research presented in this dissertation: (i) data aggregation techniques; and (ii) medium access control protocols. A brief introduction to data aggregation in sensor networks is made and previous work in this area is reviewed. MAC protocol design issues and techniques are discussed as well in Chapter 2. The relevant research literature on wireless MAC protocols is overviewed, focusing on previous work that considers timing issues at the MAC protocol layer.

Chapter 3 addresses the proposed novel protocol, WiDom, an adaptation of dominance/binary countdown protocols to a wireless channel. Chapter 3 considers the case of a SBD. The protocol design and rationale are presented and an implementation of it is demonstrated and evaluated. It is shown that the proposed protocol is collision-free, does not require synchronized clocks and supports a large number of priority levels. WiDom enables static priority scheduling in wireless systems and therefore, a response-time analysis is also proposed and tested. Finally, this chapter also describes how the reliability of the protocol can be improved by retransmitting priority bits.

Chapter 4 extends WiDom for supporting wireless networks with MBD, where the hidden node problem must be dealt with. The proposed solution is the first prioritized and collision-free MAC protocol designed to successfully deal with hidden nodes without relying on out-of-band signaling. The novel protocol is experimentally evaluated both using simulation and real-world platforms.

Chapter 5 describes the novel distributed algorithms that allow exploiting WiDom to efficiently obtain certain aggregated quantities. Solutions are provided to obtain the minimum (MIN), the maximum (MAX), MEDIAN, COUNT and Interpolation in a SBD. techniques on how to adapt to MBD the solutions proposed for a SBD are also

presented.

Chapter 6 shows that highly scalable aggregate computations in wireless networks are possible in practice. This is done by (i) building a new wireless hardware platform with appropriate characteristics enabling efficient dominance-based MAC; (ii) implementing dominance-based MAC protocols on that platform; (iii) implementing distributed algorithms for aggregate computations (MIN, MAX, Interpolation) as described in Chapter 5, using the new implementation of the dominance-based MAC protocol; and (iv) performing experiments to prove that such highly scalable aggregate computations in wireless networks are possible.

Chapter 7 summarizes the contributions presented, raises some points of discussion and directions for future work.

Chapter 2

Background

Contents

2.1 Introduction

The primary purpose of deploying a sensor network is to collect data from the environment about some phenomena of interest. It is possible to collect all sensor readings from the network, but it might be sufficient, or even better, to only collect aggregated data of these sensor readings.

Data aggregation is the combination of data (sensor readings) from different sources by using functions such as AVERAGE, SUM, MIN, MAX or suppression (elimination of duplicates) [3]. Often, data aggregation is also referred to as *data fusion*, especially in the context of applying signal processing techniques to perform data aggregation. In this dissertation, the term aggregate quantities is used because the algorithms developed are not for general data aggregation; they only allow obtaining certain aggregate quantities such as MIN, MAX, MEDIAN or COUNT.

The rest of this chapter is organized as follows. Section 2.2 provides an overview of data aggregation techniques. While the approach for obtaining aggregate quantities in this thesis differs, in essence, from most research carried on previously, it is important to discuss and survey the most important techniques to put our approach into perpective. It is also important to note that some of these techniques may be relevant when addressing networks with MBD.

Some background material on medium access control (MAC) in wireless networks is later presented in Section 2.3. The MAC defines the way computing nodes share the radio channel for communication, and its foremost aim is to avoid collisions in the medium. Because the radio channel is a limited resource shared by a large number of nodes, it needs to be managed very carefully. In this dissertation, the role of the MAC protocol assumes even greater importance, given the fact that it will be designed to enable efficient aggregate computations. In particular, Section 2.3.4 presents a familily of MAC protocols named Dominance/Binary Countdown protocols, which are the main inspiration for the protocol proposed in Chapters 3 and 4.

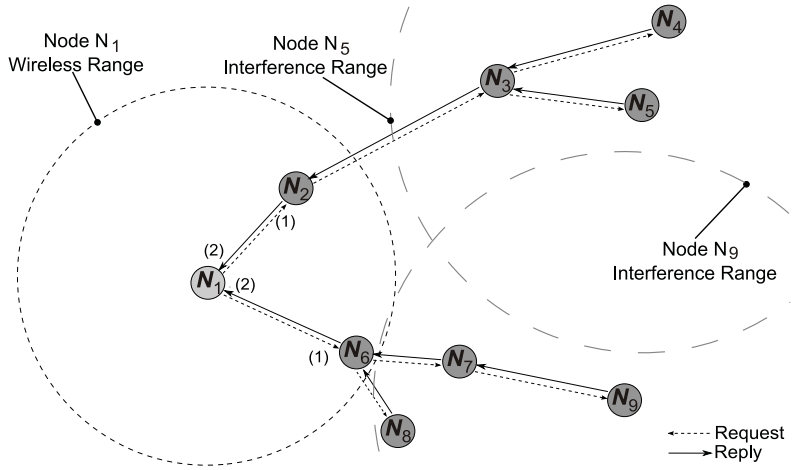


Figure 2.1: Data Aggregation Exploiting Parallel Transmissions.

2.2 Data Aggregation and Clustering

The way most data aggregation protocols achieve in-network data reduction is by allowing nodes to apply, fully or partially, aggregation functions (also called data reduction functions) on the data while it travels through the network. This often assumes that data originated at several sources flows to a sink along a tree, and thus intermediate nodes can apply data aggregation functions. Figure 2.1 presents one such scenario, where (1) a sink node, N_1, propagates its interest, and then, (2) data can be aggregated along the return path. Suppose, for example, that N_1 propagates its interest in knowing the MIN value of the temperature readings amongst all nodes. After sending the request for this data, nodes can periodically send the data back to N_1 (some works have used the notion of *epochs* to denote the period of the data transmission and aggregation functions [12]) and perform in-network processing on the data. In the example at hand, for instance, N_3 can apply the MIN to the values received from N_4 and N_5 and instead of sending both values, sends only the MIN of the two.

Observe, in Figure 2.1, that N_5 and N_9 can transmit their data in parallel. This shows that besides reducing the number of packets transmissions in the network, we are also exploiting the fact that parallel transmissions are possible. The execution

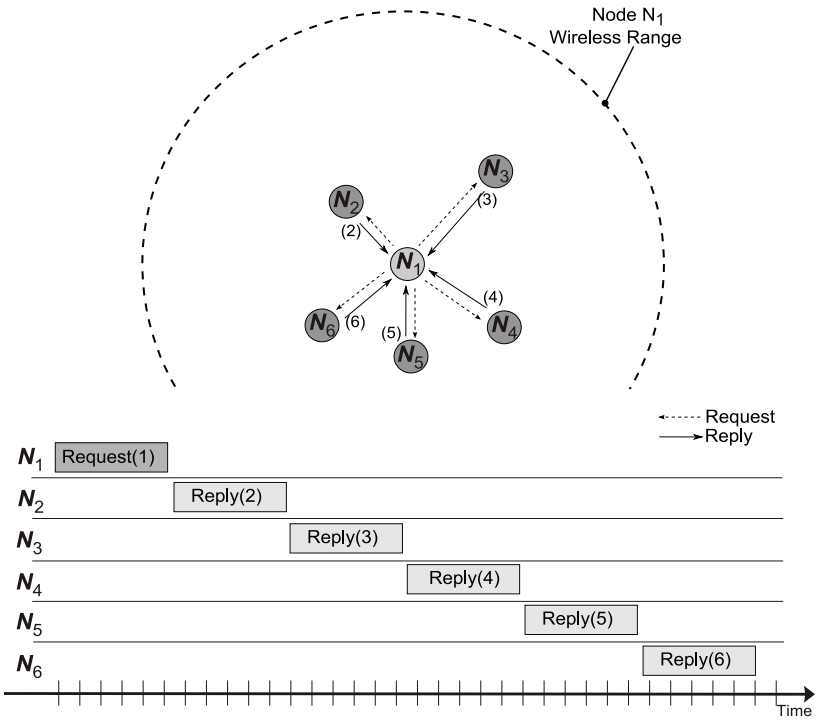


Figure 2.2: Data Aggregation in SBD (no Parallel Transmissions).

time of this algorithm will depend on the network topology and also on number of nodes.

Now let us consider the example depicted in Figure 2.2, where a node (node N_1) needs to know the MIN of the temperature readings among its neighbors, all in the same broadcast domain. One approach to this problem would imply that (i) N_1 broadcasts a request to all other nodes and then (ii) waits for the corresponding replies from them. For the sake of simplicity, let us assume that nodes set up a scheme to orderly access the medium in a time division multiple access (TDMA) fashion and that the initiator node (N_1) knows when to terminate the algorithm and compute the MIN. It is commonly accepted that *traditional* data aggregation in such scenario is pointless. We are no longer able to take advantage of parallel transmissions and, more importantly, there is no opportunity to perform data aggregation so the number of transmissions is reduced.

Under the assumption that all nodes are in the same broadcast domain, one could

think of better schemes that could effectively reduce the average number of messages being transmitted.

> *Nevertheless, it remains that all schemes discussed above have an execution time that depends on the number of nodes.*

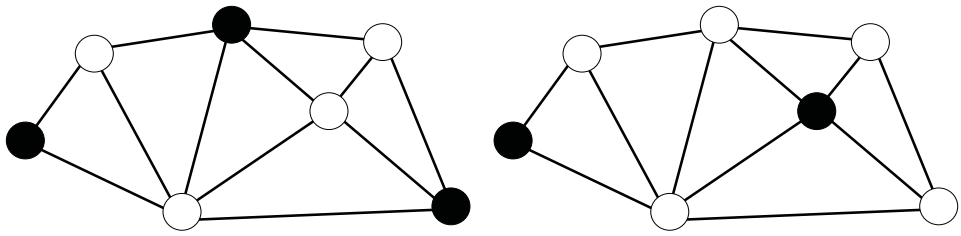
As we will see, the approach taken in this dissertation differs significantly from most research found in the literature, where most of the emphasis was put into applying data reduction functions to data coming from different sources and exploiting the opportunities for parallel transmissions.

In the following sections, some relevant related work found in the research literature is reviewed. Section 2.2.1 overviews material on clustering techniques. Previous research works approached the problem of data aggregation by grouping nodes into clusters. In this research, clustering techniques were used to group nodes into SBDs and allow techniques developed for obtaining aggregate quantities in a SBD to be used in MBD networks. Consequently, Section 2.2.2 reviews other relevant research work related to former data aggregation techniques.

2.2.1 Clustering

Several previous research works have approached the problem of data aggregation by grouping nodes into clusters. Clustering techniques have been previously exploited for load balancing, fault-tolerance, increasing connectivity or maximizing network longevity [13].

The relevance of clustering in the context of this work arrises from the fact that it is employed (in Chapter 5) to allow using and adapt the algorithms developed for the SBD to the MBD case. Essentially, this is achieved by forming clusters of nodes (which, to emphasize that each cluster is a SBD, are called *partitions* in Chapter 5) where the nodes in each cluster form a SBD. The leaders of each cluster are responsible for collecting the aggregate data in each cluster and forward it. Cluster leaders can



(a) 3 node DS. (b) 2 node DS.

Figure 2.3: Illustration of a Dominating Set – *DS* (Black Nodes Form the DS).

still apply in-network data reduction functions to data received from other cluster leaders. Such description implies that each node is either a cluster leader or is a one hop neighbor of a cluster leader. The cluster leaders form a *Dominating Set* (DS).

In order to communicate the aggregate quantities from each cluster, it is also necessary that cluster heads are assigned such that there is a connected backbone between nodes; that is, the DS must be connected. Therefore, it is necessary to find a distributed algorithm that (i) determines a *Connected Dominating Set* (CDS) and (ii) *all one-hop neighbors of each cluster head are in the same broadcast domain.* In the remainer of this section, the attention will be on the DS problem. A general survey of clustering algorithms for WSN can be found in [13], and a performance comparison of some of those algorithms is done in [14].

Dominating Sets. Dominating Sets is one of the most important graph problems [15], with an enormous range of possible applications. Generally, DS problems arise in location challenges such as optimal location of hospitals, fire stations, schools, radio stations or communication processors in computer networks [15].

More formally, a DS is:

> a subset $D \in V$ where each node in the set of all nodes V is either in the dominating set D or is adjacent to a node $d \in D$.

Figure 2.3 illustrates different possible selections on nodes in a DS. In general, it is desirable that the DS is small, or even minimum. If the DS has the minimum

cardinality, then it is said to be a to be a *Minimum Dominating Set* (MDS).

It is well known and accepted that the MDS problem is NP-hard [15]. In general graphs, it is not possible to approximate the MDS within $c\ log|V|$ for some $c > 0$. That is, there is no polynomial time algorithm that can always find a dominating set that has at most $c\ log|V|$ more nodes than a MDS. However, the MDS problem is approximable within $1 + log|V|$ [16].

If the DS is required to be connected (a *Minimum Connected Dominating Set* - MCDS), the problem is not easier (finding the MCDS is also NP-hard). The MCDS is approximable within $\Delta + 3$, where Δ is the maximum degree of the original graph [16].

The amount of literature related to the various aspects of DS is vast. Here, only a few relevant results are mentioned. The focus is placed on exemplifying a few algorithms that approximate a MCDS and then on the algorithm selected (Chapter 5), which allows to approximate a MCDS where all one-hop neighbors of each cluster head are in the same broadcast domain.

The most elementary means to approximate a MCDS with a centralized algorithm is by growing a tree [17]. That is, iteratively adding nodes and edges to the tree. At each iteration, the two-hop neighborhood is scanned to find the pair of nodes that cover the biggest number of other nodes (this is called the *yield*) if inserted in the DS. This method was implemented in a distributed fashion to approximate an MCDS used to facilitate routing in ad-hoc networks [18].

Another centralized approach to aproximate a MCDS is to first select a MDS and then connect the elements [17]. The idea is to iteratively select the nodes in the DS that reduce the most the number of nodes left to be covered. Then, in a second step of the algorithm, the nodes in the MDS are connected recursively. This algorithm is also implemented in a distributed fashion [18].

The algorithms described previously may lead to a significant number of transmitted messages. An alternative approach is to try to find a DS that is eventually much larger than the minimum, trying to exchange a small amount of messages and then prune redundant nodes from the CDS [19]. Based on that pruning approach, a new

heuristic to remove redundant nodes was proposed [20].

Many other MCDS approximation algorithms can be found in the literature. Nevertheless, as described in the beginning of this section, it is necessary to find an algorithm that approximates a MCDS and guarantees that all one-hop neighbors of each cluster head are in the same broadcast domain. The only algorithm found in the research literature that can be adapted to do this is the one proposed in [21]. The following paragraphs describe it.

Minimum Virtual Dominating Set (MVDS). An algorithm developed for topology retrieval at multiple resolutions in large-scale dense sensor networks was introduced in [21]. This algorithm approximates the solution for a *Minimum Virtual Dominating Set* (MVDS). *Virtual* because a virtual range is used, thus the dominating set in constructed in a virtual graph (note that this MVDS is still connected, but that designation was dropped for simplicity). The virtual range is used to control the resolution of the topology information retrieved by the algorithm. In the context of this thesis, it is used to guarantee that all nodes in a partition are in the same broadcast domain by defining this virtual range as a function of the radio broadcast range (see Chapter 5).

It is a distributed algorithm with a *propagation phase* that forms the partitions and colors the nodes according to their functionality (*black* if the node is a partition leader or *red* if it is a slave member of a partition). There is a *response phase*, where the topology information is delivered to the leader node. In the beginning of the algorithm all nodes are *white*. The node starting the algorithm (the leader) colors itself black and broadcasts a message with its color. Nodes within the virtual range of the black node become red and nodes that receive the broadcast but are outside the virtual range become blue (distances can be approximated; e.g, using the signal strength from received packets). After a time interval that is inversely proportional to the distance from the black node, both red and blue nodes forward the message, if they have not done so. Upon being colored, all blue nodes start a timer to become black. This algorithm approximates the solution for a MVDS(r) composed of the nodes colored

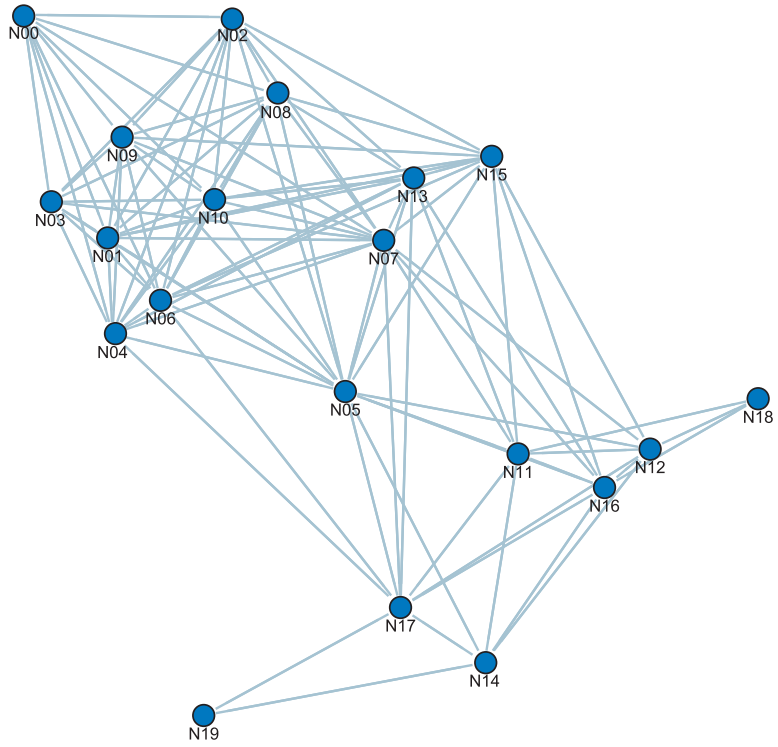


Figure 2.4: Network Topology for Figure 2.5.

black, where r is the virtual range used.

A possible selection made by the algorithm is illustrated in Figure 2.5. Figure 2.4 presents the topology of the network. The different partitions formed are depicted in Figure 2.5 by representing the nodes in the same partition similarly. Figure 2.5 also depicts the partition leaders selected (with a circle around the node) by the algorithm and their respective virtual ranges.

2.2.2 Other Relevant Previous Work on Data Aggregation

One early work on data aggregation is Directed Diffusion [22]. It is a data-centric protocol where data generated by nodes is named by attribute-value pairs. Nodes start by broadcasting their interests for data and these interests are flooded to the whole network. Data matching these interests is then delivered towards the node. Intermediate nodes can perform data caching and transformation to achieve robust

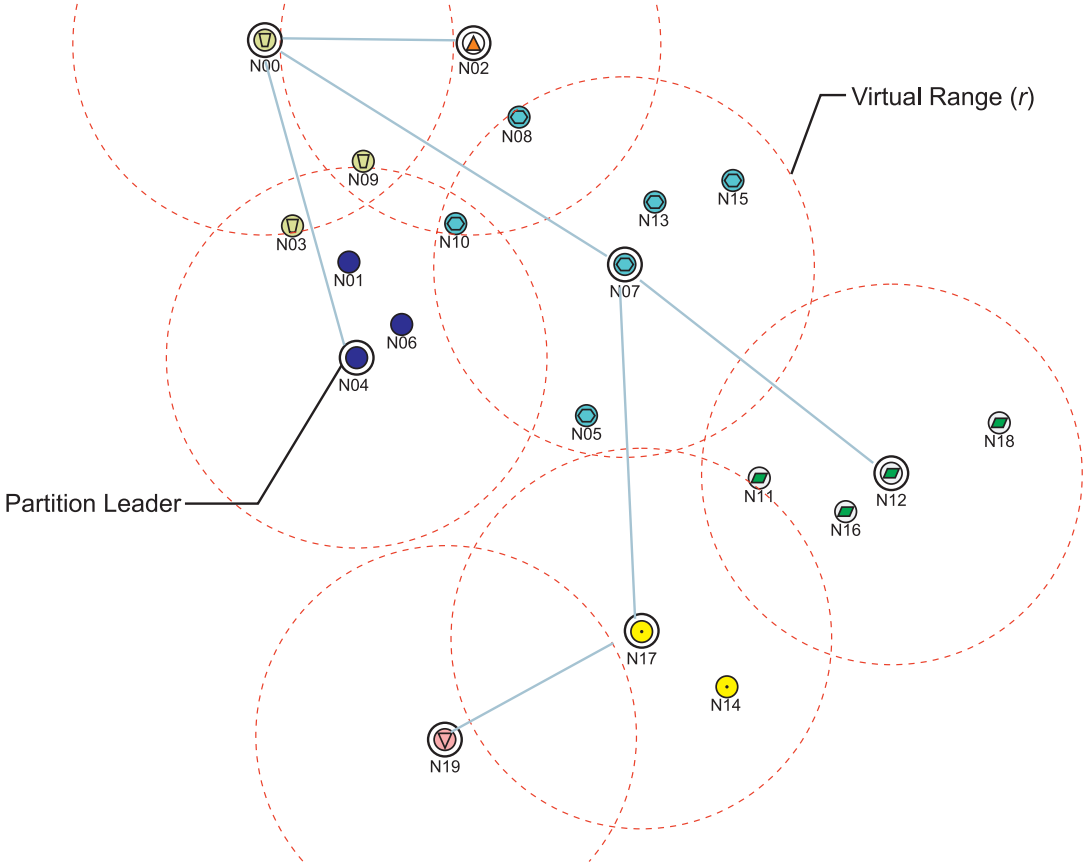


Figure 2.5: Illustration of the MVDS(r) Construction Algorithm.

data delivery, coordinated sensing and data reduction and directing interests.

Data aggregation protocols have focused on tree-based or cluster-based structured approaches [23, 24, 12, 25, 26]. The Low energy Adaptive Cluster Hierarchy (LEACH) [23] lets nodes organize themselves into clusters, where one node acts as a cluster head. All non-cluster head nodes transmit their data to the cluster head. The cluster head can, while receiving the data from nodes in the cluster, apply data reduction and compression functions on the data and transmit the results to a base station (BS).

Another work, Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [24], organizes all nodes in a chain so that each node transmits to and receives from only one closest node of its neighbors. Nodes perform the role of heads in turn,

to conserve energy. This is done by randomly choosing a node from the chain that will transmit the aggregated data to the BS, thus reducing the per round energy expenditure as compared to LEACH. However, in PEGASIS, latency is an issue. Only one node is allowed to transmit to the BS at each time, and long delays can be introduced for nodes distant on the chain.

Both LEACH and PEGASIS assume that the BS can be reached in one hop, which, in practice, can be a very restrictive assumption.

One way to use aggregation algorithms is to encapsulate them in a query processor for database queries. The advantage of such approach is that it can decouple the logical view of the data from the actual implementation of accessing to data. Query processors for sensor networks have been studied in previous works such as TinyDB [12] and COUGAR [25]. They assume one single sink node and that the other nodes should report an aggregate quantity to this sink node. The sink node floods its interest in the data it wants into the network and this also causes nodes to discover the topology. When a node has new data, it broadcasts this data; other nodes hear it, then it is routed and combined so that the sink node receives the aggregated. These works exploit the broadcast characteristics of the wireless medium but they do not make any assumption on the MAC protocol (and hence they do not take advantage of the MAC protocol).

One important aspect of these protocols is to create a spanning tree. It is known that computing an optimal spanning tree for the case when only a subset of nodes can generate data is equivalent to finding a Steiner-tree, a problem known to be NP-hard (the decision problem is NP-complete, see page 208 in [27]). For this reason, approximation algorithms have been proposed [28, 29]. It is interesting to note that, in the average case, very simple randomized algorithms perform well [30].

Since a node will forward its data to the sink using a path which is not necessarily the shortest path to the sink, these protocols may cause an extra delay. Hence, there is a trade-off between delay and energy-efficiency. To explore this trade-off, a framework based on feedback was developed for computing aggregated quantities [31]. Tree

structures are fragile under transmission failures, common in WSN. Packet loss may lead to loss of data from an entire subtree. Synopsis Diffusion [26] improves this by designing aggregation schemes that are insensitive to message duplication or message arrival order. In this way, decoupling of aggregation from message routing is achieved, allowing the routing layer to employ, for example, multi-path routing schemes.

Because, in dynamic environments, tree-based or cluster-based structured approaches suffer from high setup and maintenance overheads, some work has been developed to perform data aggregation in structure-free networks [32, 33].

Common to all the works previously mentioned is that the time-complexity increases with the number of sensor nodes. This is an significant drawback, especially in the case where a SBD contains a large number of nodes.

2.3 Medium Access Control

One of the foremost characterizing features of Wireless Sensor Networks (WSN) is the large number of nodes that must work collaboratively to achieve some goal. In the near future, it is expected that the evolution of processing, storage and transmission of information in wireless networks will enable the construction of WSN with thousands of very small and inexpensive nodes [34] (each node having tenths of nodes within communication range, dimensions in the order of a few cubic millimeters and individual nodes cost almost negligible). In this context, the Medium Access Control (MAC) protocol assumes a determinant role in the global performance of the WSN. The MAC defines the way computing nodes share the radio channel for communication, and its foremost aim is to avoid collisions in the medium. Because the radio channel is a limited resource, shared by a large number of nodes, it needs to be managed very carefully. Furthermore, if we observe that communication is the most expensive operation a node performs in terms of energy usage [34], being the MAC protocol one of the most relevant sublayers involved in this process, the role of the MAC protocol is even more emphasized.

Several key characteristics of wireless sensor networks justifying the research of novel MAC protocols have been identified [35]. These characteristics are now presented and briefly discussed at light of recent developments since that work was published:

Collaborative nature of WSN. In WSN, nodes typically have to work collaboratively to serve one or a small number of applications. At a given point in time, a node may have more relevant data for the system as a whole, and, for this reason, contrarily to more traditional communication systems, fairness at the node level becomes less relevant than overall application performance.

Sporadic information processing and delivery. Many WSN applications are designed to respond to stimuli from the environment. Typically, these events are triggered sporadically; requiring that nodes stay idle most of the time. When event(s) of interest occur, this causes a burst of activity in the network. Furthermore, as identified in [36], this activity is often spatially-correlated due to the simultaneous detections of the same event by nodes in the same neighborhood. This sporadic nature often imposes that applications are prepared to deal with large latencies. However, a growing number of WSN applications cannot cope with such latencies (e.g. [37]).

In-network processing. Instead of blindly forwarding all data, nodes can perform processing of the data received and avoid spurious transmissions. Examples of such techniques have been discussed previously in Section 2.1. Such techniques make no assumptions about the MAC protocol used, and thus do not take advantage of it. However, it has been shown that exploiting the properties of the MAC protocol can greatly reduce the number of messages necessary for performing distributed computations or gathering certain aggregated quantities [38, 39].

Lack of mobility. It is often accepted that in most WSN applications, nodes are static, and thus the MAC protocol can be designed to exploit the relatively static neighborhood of the nodes. However, different factors may hinder the effectiveness of such strategy. For example, radio irregularities cause connectivity between nodes to change even though nodes are static [40]. While researchers have pointed out that these are issues mainly affecting the routing layer [40], the MAC protocol must be able

to gracefully deal with these issues. Furthermore, recent projects make use of mobile sensor nodes, for example in medical care and disaster response applications [41].

Energy efficiency, Scalability and Robustness. These issues are of paramount importance in WSN and often designers trade-off standard protocol objectives like fairness or latency for the sake of network lifetime. Because most WSN applications are deployed in an ad-hoc manner and operate in unpredictable environments, scalability and adaptability to changes in size, density and topology are relevant features to take into account in the design of MAC protocols.

2.3.1 The Design Space of MAC Protocols

There are some common strategies for sharing a communication channel (either wired or wireless) that are also relevant to the context of WSN because they can serve as a reference to common designs.

These strategies include time division multiple access (TDMA) schemes, where messages are assigned to time slots in a way that no two nodes transmit at the same time. Typically, these communication protocols operate on the basis of TDMA cycles, where a node is assigned one or many time slots. Usually, each slot has a fixed length and the number of slots per cycle is also fixed. Hence, a TDMA cycle has fixed and known time duration, and upper bounds on messages' queuing delays can be proven. Examples of such protocols can be found in the context of wired systems [42, 43, 44], in wireless systems where, for example, they are combined with a frequency division scheme in the widely used GSM protocol [45] and also in the context of WSN [46, 47], which we will discuss later on.

Another option is the use of the well known strategy called carrier sense multiple access (CSMA). Several different versions of CSMA have been developed. For wired networks, Carrier Sense Multiple Access/Collision Detection (CSMA/CD), used in Ethernet networks, is the most popular one. The general idea of the CSMA/CD MAC protocol can be described in the following way. When a station requests to transmit, it

listens to the cable. If the cable is busy, the station waits until it goes idle. Otherwise, it transmits immediately. If two or more stations begin transmitting on an idle cable simultaneously, the messages will collide. All colliding stations then terminate their transmission, wait a random time, and repeat the whole process all over again.

This simple approach cannot however be used in wireless networks, as there is no means to directly detect collisions of ongoing transmissions due to the large difference in transmitted and received energy. Therefore, other approaches have been developed. The first MAC protocol developed for wireless data communication was ALOHA [48], where nodes do not sense the channel (i.e. listen to the channel) before transmitting. In ALOHA, messages were transmitted when required so, and acknowledgements were expected to confirm correct transmissions. ALOHA had the advantage of being very straightforward, but the maximum channel utilization was very low (only about 18.6% of the channel capacity [48], or 36.8% using the slotted version). For this reason, CSMA was introduced in wireless networks [49]. In CSMA, nodes sense the channel before transmitting, and defer transmission if the channel is busy. CSMA performs significantly better than ALOHA in SBD networks (that is, a network where every node is able to perceive every transmission). In networks with MBD (networks where a single transmission cannot reach all nodes), CSMA does not perform better than ALOHA because, in such networks, sensing the channel activity at the transmitters does not convey any information about the state of the channel at the intended receiver, and thus, a CSMA technique alone cannot prevent collisions at the receiver. This is a well-know phenomenon originally labelled *hidden terminal problem* [50]. In this dissertation we employ the term "hidden node problem" to refer to this phenomenon. The hidden node problem and other relevant challenges to the design of MAC protocols for wireless networks are overviewed in Section 2.3.2.

Another strategy which cannot be trivially applied to wireless networks is the principle implemented in a family of MAC protocols called *dominance protocols* or *binary countdown protocols* [4]. These protocols are implemented in the Controller Area Network (CAN) bus [5], a wired bus network used pervasively in various industries such

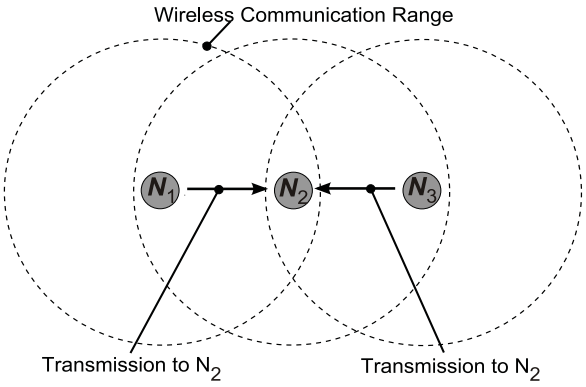


Figure 2.6: Example Illustrating Hidden Nodes.

as the automotive industry, building automation or process control. It is a prioritized MAC protocol where nodes contend for the medium by sending a unique ID (which acts as a priority) bit-by-bit. After sending all priority bits, only one node will be elected to send its data. The principle behind dominance/binary countdown protocols is the main inspiration for the protocol proposed in Chapters 3 and 4, and it will be presented in more detail in Section 2.3.4.

2.3.2 Wireless MAC Challenges

The hidden node problem and other challenges imposed by the ad-hoc nature of some wireless networks (such as WSN) are important problems to the design of MAC protocols. The following sub-sections will briefly address the hidden node, the exposed node and the priority inversion problems, along with approaches developed to mitigate them.

2.3.2.1 The Hidden Node Problem

A node is said to be hidden from another node if those nodes are out of each others' range, while both are within the range of a third node. Because the two nodes (e.g., N_1 and N_3 in Figure 2.6) cannot detect when the other is transmitting, they may cause undue collisions at a third (receiving) node (N_2 in Figure 2.6).

The hidden node problem has received serious attention in the research community because it can cause collisions which lowers system throughput, and for time-sensitive traffic, dealing with hidden nodes is even more crucial, since a collision may cause a deadline miss.

The notion of hidden nodes was reported in the mid seventies and a solution, called the busy tone solution, was proposed and analyzed [50]. In that work it was assumed that nodes communicate with a base station, and the base station can transmit to all nodes. Whenever a node transmits, the base station hears a carrier wave and the base station transmits a tone on a narrow band to all nodes. This prevents the hidden node problem. In cases where there is no base station, a similar scheme was proposed called receiver initiated busy-tone multiple access [51]. In that approach, an ordinary node (which is not a base station) listens for a carrier wave and when it hears a carrier wave, it transmits a busy tone on a narrow band channel when it is receiving data (or has heard one request to send), hence informing other senders that they should not transmit. Common to these two solutions (reported in [50] and [51]) is that they rely on a separate channel. To remove this limitation, MACA was later proposed [52].

In MACA, a node that requests to send a data packet sends a Request-To-Send (RTS) packet on the channel (there is only one) and then wait for the receiving node to transmit a Clear-To-Send (CTS) packet to the sender's ID. Then, the node can transmit; all other nodes are not permitted to transmit. Certain parameters were left unspecified in MACA, thus a more well-defined protocol called MACAW was proposed [53]. If, however, the receiver initiates communication then the RTS can be dropped; hence, MACA-BI (MACA by invitation) was developed [54]. In MACA-BI, communication starts with the receiver inviting a sender with a CTS. MACA-BI offers a lower overhead than the RTS/CTS dialogue (in MACA), but since it depends on the receiver polling senders, it is inefficient for the case when a node transmits sporadic messages.

A common weakness of these protocols is that collisions may occur, even from non-hidden nodes. To rectify this situation, a medium-access control protocol without

collisions was designed and given the name FAMA-NCS [55]. It is based on (i) a RTS/CTS dialogue but also on carrier sensing before transmission of RTS and CTS and (ii) carefully selected lengths of these messages and intervals of silence. This ensures that collisions between RTS and data packets cannot occur. Schemes with only RTS/CTS (and with no carrier sensing) have been explored as well. In order to ensure that a data packet does not collide with an RTS, it is necessary that the receiving node sends a number of CTS which is large enough to be sure that all neighbor nodes of the receiving node have received collision-free CTS. This implies that if carrier sensing is not used then a sender needs to wait for a long time before it can transmit a data packet. Hence, it was concluded that carrier sensing is necessary to implement collision-free MAC protocols efficiently [55]. The CTS packet has the role of the receiver informing the sender that it is OK to transmit. Another way to achieve the same effect is to let the receiver be idle for a time duration if it is OK to transmit; if the receiver (or other nodes) detected a collision between RTS packets then the receiver (or other nodes) transmits a jamming signal. If the Tx-Rx turnaround time (that is the time to switch from transmitting to receiving and vice-versa) is larger than the time-of-flight (which is common) and there is only one sender and one receiver and no other nodes in the network, then this solution does not work.

Another solution to achieve collision-free transmission of data packets in the presence of hidden nodes is to use the RTS/CTS dialog with no carrier sensing (just like MACA) but also use two busy tones on separate channels. A tone (called BTt) is transmitted by the sender when it transmits a data packet and another tone (called BTr) is transmitted when the receiver receives a data packet. The authors proved that this solution is collision-free [56]; it offers high throughput [57] and it also has the advantage of solving the problem of exposed nodes (addressed later in Section 2.3.2.2). Nevertheless, this solution requires extra bandwidth for the two busy tones and specialized hardware to detect those busy tones.

The FAMA protocol ensures that data packets are collision-free but it is still possible for RTS packets to collide if two RTS packets are transmitted from two different

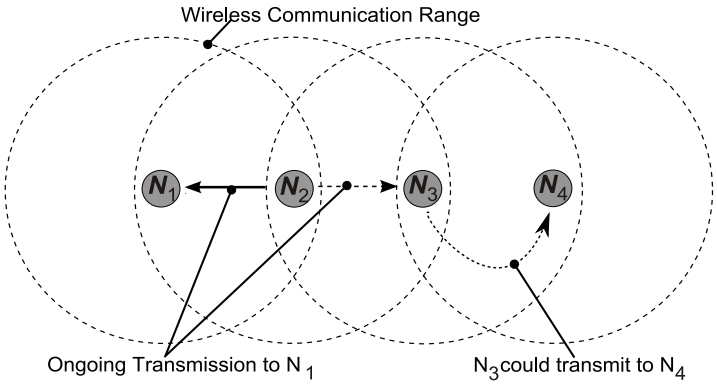


Figure 2.7: Example Illustrating the Occurrence Exposed Nodes.

nodes and the difference between their transmission times is less than the time-of-flight. In very high loads, this can cause the channel to only transmit colliding RTS packets and hence the throughput of data packets drops to zero. To rectify this, a protocol with collision resolution was proposed [58, 59]. It works as follows. When a node transmits a RTS and it does not hear a CTS it concludes that the RTS collided. Then, only half of all nodes are permitted to transmit the next RTS; the other half of nodes must wait. These groups are selected based on the IDs of the nodes. By repeating this procedure (called tree-splitting) eventually only one node transmits the RTS.

Both hidden and exposed nodes were also studied in [60]. The novel approach in that work was to address hidden and exposed nodes together, whereas previous work typically handled both problems separately. An interesting conclusion of was that there is generally a tradeoff between hidden and exposed nodes, and entirely eliminating both of them together appears to be difficult.

Instead of avoiding hidden nodes, another work [61] focused on exploiting asynchrony across successive collisions. The main idea was that there would be collision-free parts of the packets in each collision and these could be used to correctly decode packets. While reducing substantially the number of lost packets due to hidden nodes, with this scheme collisions are not avoided and, most importantly, the number of collisions is unbounded.

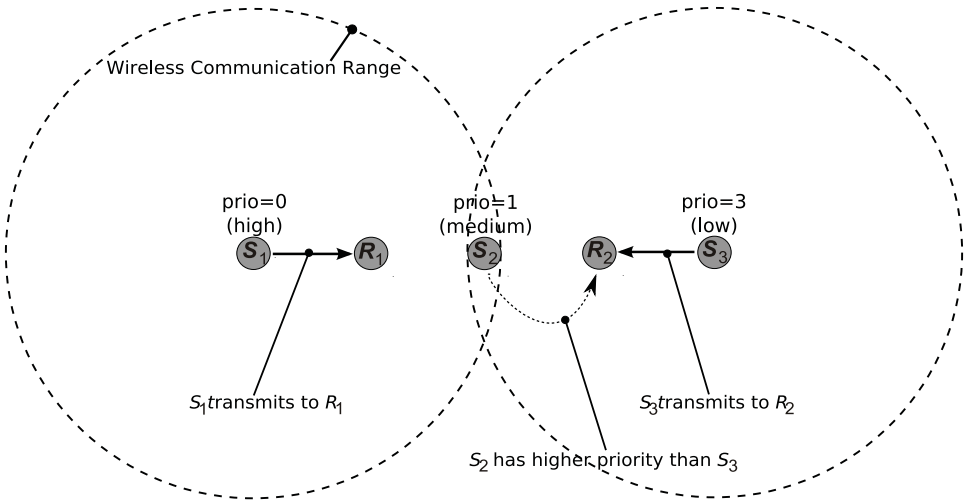


Figure 2.8: Example of Pseudo Priority Inversion.

2.3.2.2 The Exposed Node Problem

Exposed nodes occur when a node (N_3 in Figure 2.7) refrains from transmitting because a neighbor node transmits (N_2), but the receiving nodes are far apart and do not experience a collision, e.g. N_1 and N_4. In the scenario depicted in Figure 2.7, N_3 is said to be an exposed node to N_2; and conversely, N_2 is said to be an exposed node to N_3. The existence of exposed nodes may reduce the number of parallel transmissions, but it does not violate the correctness of reception. For this reason, exposed nodes are generally not considered to be as severe as hidden nodes. Nonetheless, several attempts to solve it exist. MACA and MACAW [52, 53] provide a partial solution; the dual busy tone solution [57] is a fully functioning solution, and [62] attempts to find out the sender/receiver roles of different nodes to achieve even more parallelism.

2.3.2.3 The Pseudo Priority Inversion Problem

Priority inversion means that at least two messages are awaiting transmission but the lowest priority is transmitted before the highest priority message. This problem is well-known in uniprocessor scheduling (where tasks are prioritized rather than messages) [63]. In real-time communication, the same problem may occur due to the

non-preemptive nature of communication, so it is an issue in multihop networks too.

However, in multihop networks, yet another, but related problem may occur [64]. Consider Figure 2.8, where three sending nodes S_1, S_2 and S_3 and two receiving nodes R_1 and R_2 are illustrated. All three sender nodes request to send at the same time: S_1 requests to send to R_1; S_2 and S_3 request to send to R_2. S_2 is within S1s range. S_1 and S_2 compete for R_1. S_2 and S_3 compete for R_2. S_1 has higher priority than S_2 which has higher priority than S_3. S_1 will transmit because it has the highest priority. S_2 cannot transmit because it lost the competition against S_1. S_3 will transmit although it competes with S_2 which has higher priority; this is because S_2 lost the competition versus S_1. This condition is called *pseudo priority inversion* and it was also shown that this effect can cascade, causing chains where a message is dependent on another message arbitrarily far away [64].

2.3.3 Timeliness-Aware Wireless MAC Protocols

The review of MAC protocols developed for the broad area of wired and wireless data communication is a very challenging task, mostly due to the plethora of previous research in the area. This is indeed not the role of this section. While focusing on previous research for wireless communication does reduce the work to be covered, it would still be a daunting task. Furthermore, several surveys [65, 66, 67] of MAC protocols for wireless data communication exist in the literature.

Restricting the scope to MAC protocols specifically designed for WSN does not make this task substantially easier. The research literature on MAC protocols for WSN is vast, and researchers have addressed different design goals such a fairness, low power consumption or high throughput. Surveys that cover the topic of MAC protocols for WSN can be found in [34, 68, 69].

In this section, we will focus instead on previous work that dedicated attention to the development of MAC protocols with some concerns related to timeliness. Researchers have been exploring problems related to supporting messages with deadline

requirements in wireless ad-hoc networks long before the establishment of the WSN research area. Therefore, it is important address some of the results related to those works.

IEEE 802.11-Based Protocols. The introduction of the wireless LAN standard IEEE 802.11 stimulated the development of many prioritized CSMA protocols. Some of these protocols [70, 71, 72] changed parameters in the IEEE 802.11 standard to be a function of deadlines, either choosing (i) inter-frame spacing (the amount of time that a node waits before transmitting) or (ii) the back-off times after a collision has occurred. These techniques are useful to meet deadlines because they can implement algorithms such as deadline monotonic [73]. Another work [74], piggybacks priority information of a node's head-of-the-line packet onto handshake and data packet. Using this information, nodes try to approximate priority scheduling by controlling the parameters of the back-off scheme in IEEE 802.11. These techniques [70, 71, 72, 73, 74] have two main drawbacks (i) they only approximate priority scheduling; it may happen that a high-priority message has to wait for one or many lower-priority messages and (ii) collisions can occur hence causing deadline misses. It is important to note that the IEEE 802.11e profile, introduced with the intention of offering better support for Quality-of-Service, adopted the approach of choosing back-off times as a function of priorities [70, 71, 72].

Another work [75] modifies the IEEE 802.11 protocol to avoid collisions and the transmission of packets whose deadline has already expired. It achieves this by assigning a deadline to packets; when packets are queued for transmission, a timestamp is recorded locally, if the deadline expires, the packet is dropped immediately. When the packet is about to be actually sent out, the sending node chooses the next backoff value and sends this information in the packet header. Neighboring nodes can then choose a different backoff value and thus avoid collisions. The approach reported in [75] can achieve drastic reductions on mean packet delays, missed deadlines and packet collisions. However, because the range of values from which the backoff value is chosen is

made as a function of the number of nodes in the system, the contention window may grow considerably in a network with a large number of nodes.

The IEEE 802.11 standard also defined another MAC protocol where a base station polls a node, and gives it the right to transmit in a time interval. This scheme was recently refined with traffic classes in the IEEE 802.11e profile. Naturally such approach is inefficient to schedule sporadic messages.

The concept of "black-bursts" was designed in the context of 802.11 [76, 77, 78], however they do not only change some parameters in the IEEE 802.11. They also require other signals to be transmitted. If the channel is idle then a node transmits a message immediately. Otherwise, the node waits until the channel becomes idle and transmits a "black-burst" (a jamming signal) for a time duration which is proportional to the priority. When a node finishes transmitting its jamming signal, the node listens to find out whether other nodes transmit a jamming signal. If so, the node did not have the highest priority and so it waits until the channel is idle again. The protocols based on "black-burst" were originally used to ensure that all real-time traffic was given a higher priority than non real-time traffic and dynamically change priorities of real-time traffic to achieve round-robin scheduling [77, 78]. These schemes ([77, 78]) have the following drawbacks (i) collisions can occur if the channel is idle and two nodes request to transmit simultaneously and (ii) the maximum length of the black-burst is proportionate to the number of priority levels, so only a small number of priority levels can be supported.

Another technique [79] is to implement prioritization using two separate narrow band busy-tones to communicate that a node is backlogged with a high-priority message. This technique has the drawback of requiring specialized hardware (for listening to the narrow band signals), requiring extra bandwidth (for the narrow band signals) and supporting only two priority levels. We believe that this out-of-band signaling solution [79] can be extended to k priority levels (although the authors do not mentioned it) but doing so would require $2k$ narrow band signals.

Selected Wireless MAC Protocols for WSN. We will now survey and discuss some selected Wireless MAC protocols designed specifically for WSN with timeliness concerns.

Most of the MAC protocols developed to support deadline requirements in the context of WSN are designed around some TDMA-based scheme. Indeed, TDMA protocols have very appealing characteristics for this context, such as being inherently collision-free, having the possibility of scheduling transmit/receive times, and consequently being very power efficient.

Common to all TDMA-based protocols is the requirement that nodes have the same time reference. This has been solved in a number of ways. The simplest approach is to use the Global Positioning System (GPS) as the source of a global clock. GPS can provide extremely accurate timing, but requires special (typically power hungry) receivers and a clear sky view. Nevertheless, GPS may become standard in designs of sensor network platforms in the near future.

Another approach to have a global clock is to have nodes receiving time reference synchronization beacons that reach the whole network. One way to implement that is to use a platform that supports receiving an out-of band signal. Such platform is described in Section 6.3, and other examples can be found in the literature, such as the FireFly sensor platform [80] that is equipped with an Amplitude Modulation (AM) receiver that detects time-sync signals with a continental-wide coverage.

Often, the synchronization problem is tackled by transmitting in-band synchronization information. Typically, these involve creating some form of hierarchical organization and use it to distribute timing information. There are several in-band time synchronization schemes in the research literature, where some of the most salient of these, providing good accuracy, are RBS [81], TPSN [82] or FTSP [83]. Notably, the work in [84] is the only practical synchronization strategy that does not require nodes to construct a hierarchical organization, but it can take an unbounded number of broadcasts to achieve synchronization.

While researchers have tried to mitigate some shortcomings, often TDMA-based

approaches organize nodes in clusters or cells and have a master node proving central coordination, and thus are inflexible to changes in the network topology and the number of participant nodes. Furthermore they have the drawback of requiring that sporadic message streams are dealt with using polling, which is inefficient, especially when the deadline is short, compared to the minimum inter-arrival time of the messages.

The Traffic-Adaptive Medium Access (TRAMA) protocol [85] is a TDMA-based MAC protocol that constructs schedules in a distributed manner and on an on-demand basis. It supports both scheduled slots and CSMA-based contention slots for node admission and network management, and avoids the assignment of time slots to nodes with no traffic to send. It also allows nodes to determine when they can become idle and not listen to the channel using traffic information. Unfortunately, TRAMA can consume significant computation and memory resources, since it needs to maintain and perform computations upon the two-hop neighborhood list of a node, and this can be very large in dense WSN.

PEDAMACS [86] is another relevant TDMA-based approach. This protocol is based on the existence of a single node that can reach all nodes in the network (called AP in the paper) with a single hop. Nevertheless, the protocol can be extended to deal with nodes outside the range of the AP, at the cost of increased overhead and delay. The protocol is divided in four phases: topology learning, topology collection, scheduling, and adjustment. During the topology learning phase, each node discovers its neighbors, interferers and parent nodes. Then, at collection phase, this information is gathered by the AP, that during the scheduling phase uses this information to create a TDMA schedule for all nodes. The adjustment phase is executed on demand to learn new local topology information that might not have been discovered or that has changed. The authors address the case of sporadic message streams only to state that the protocol handles this type of message streams at the cost of wasting bandwidth (time slots assigned to nodes are not used).

In [46], a hardware platform was developed to support a TDMA protocol that can

use an out-of-band synchronization mechanism to provide a global clock, avoiding in-band solutions that reduce network performance. In [87], the authors have explored the maximization of parallel transmissions over a TDMA network using RT-Link. This provides optimal end-to-end throughput by identifying the maximal set of concurrent transmitters across the network, while maintaining a bounded delay. However, this result is achieved by assuming that nodes are deployed in a regular structure, something often not applicable in practice.

Another approach, Implicit Earliest Deadline First (I-EDF) [47], is based on the assumption that all nodes know the traffic on the other nodes that compete for the medium and all these nodes execute the EDF scheduling algorithm. If the message selected by the EDF scheduling algorithm is in the node's queue of outgoing messages, then the node transmits this message, otherwise it does not transmit. Unfortunately, this algorithm is based on the assumption that a node knows the arrival time of messages on other nodes, thus nodes must be placed in static cells, and channel assignment needs to be carefully handled to avoid interference between neighbor cells. This imposes a significant limitation in the real-world applicability of this protocol, and also implies that polling must be used to deal with sporadic message streams.

The Dual-mode real-time MAC protocol [88] has two operating modes: protected and unprotected. The unprotected mode is used while no collisions are detected, after which, the protected mode is started. The protected mode is a TDMA scheme, also based on a cellular structure, where each cell has a different channel. In this respect, it is very similar to Implicit EDF, and thus it has the same drawback of static cells and requires careful channel assignment.

The IEEE 802.15.4 [89] standard covers the physical and MAC layers of a Low-Rate Wireless Personal Area Network (LR-WPAN). It is important to distinguish the IEEE 802.15.4 standard from ZigBee [90]. ZigBee is an industry consortium with the goal of ensuring interoperability between devices. It uses the services provided by IEEE 802.15.4 and defines the higher networks layers and application interfaces to do so. IEEE 802.15.4 was designed for deployment of low-cost, low power wireless networks

able to run for years at very low duty cycles.

The MAC layer in IEEE 802.15.4 has several operating modes. For the purpose of this section (supporting messages with deadline requirements in WSN) the most interesting mode is the beacon-enabled mode, where nodes organize themselves in a Personal Area Network (PAN), and a PAN coordinator organizes channel access and data transmissions in a structure called the superframe.

The PAN coordinator is responsible for periodically transmitting a beacon frame announcing the start of the superframe. The superframe is divided into two main periods: the active period and the inactive period. During the inactive period, nodes in the PAN can turn off their radios, to save energy. The active period is subdivided into 16 time slots, where the first time slot (slot 0) is reserved for the beacon frame. The remaining slots (1 to 15) are used for the Contention Access Period (CAP) and for a maximum of seven Guaranteed Time Slots (GTS). During the CAP, nodes access the medium using slotted CSMA/CA (CA stands for Collision Avoidance), whereas the GTS is used for reservation-based TDMA access. The GTS slots are allocated by the PAN coordinator, and nodes perform reservation requests during the CAP. A thorough review of IEEE 802.15.4 in the context of supporting messages with deadline requirements in WSN can be found in [91].

A performance study of slotted CSMA/CA can be found in [92], and [93] introduces a mechanism for service differentiation in slotted CSMA/CA by simple manipulation of the protocol's parameters according to the priority of messages. The GTS allocation mechanism was also subject of several studies that address the throughput and delay guarantees provided by this mechanism [94], and energy/delay trade-offs [95]. To overcome the maximum limit of seven GTS allowed, in [96] the authors propose i-Game, an implicit GTS allocation mechanism that enables the use of a GTS by several nodes.

The Bluetooth [97] system was designed for a WPAN, with one major application in mind: The connection of mobile devices to a personal computer. Bluetooth organizes nodes into piconets with one master and only up to seven slave nodes. Bluetooth was already employed for prototyping WSN [98], but, the fact that it needs a master

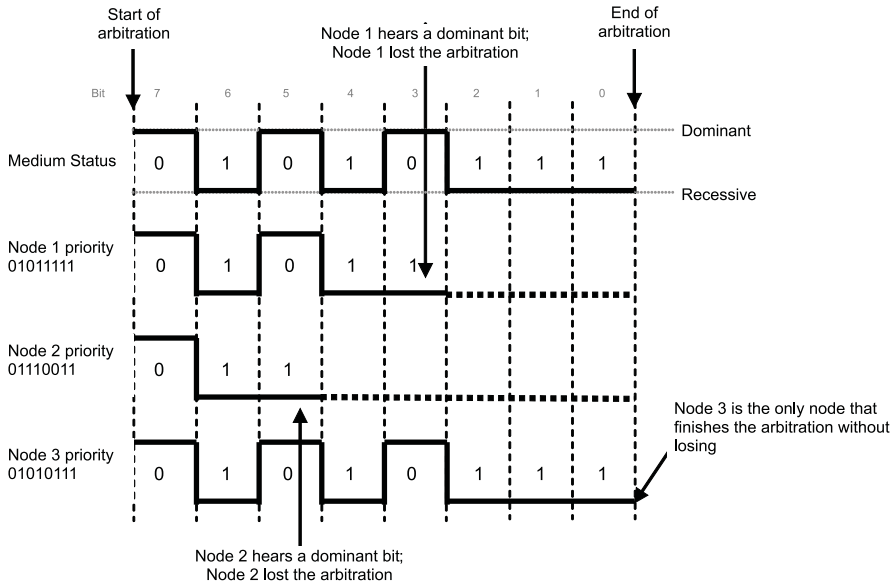


Figure 2.9: Arbitration in Dominance/Binary Countdown Protocols.

continuously polling its slaves, and the relatively low number of supported slave nodes limits its application in WSN.

2.3.4 Dominance/Binary Countdown Protocols

In the implementations of Dominance/binary countdown protocols [4] (e.g., the Controller Area Network – CAN [5]), messages have a unique contention field, which corresponds to a unique priority that is used to resolve the contention for channel access. A node that requests to transmit waits for a pre-determined time interval until the channel is idle. Then it starts a conflict resolution phase – the arbitration – where each node sends its contention field bit-by-bit, starting with the most significant bit, while monitoring the medium at the same time. The medium must be devised in such a way that nodes will only detect a recessive bit if no node is transmitting a dominant bit. If any node is transmitting a dominant bit, then every node will detect a dominant bit regardless of what the node itself is sending. During the arbitration, if a node contends with a recessive bit but hears a dominant bit, then it will refrain from transmitting any further bits and will only monitor the medium. Finally, only one node (assuming

that the contention field is a unique priority) reaches the end of arbitration without hearing a dominant bit, and therefore will proceed with transmitting the data part of the message.

The arbitration performed in dominance/binary countdown protocols is illustrated through an example in Figure 2.9. Three nodes with different priorities contend for the channel. If a bit is "0" then it is dominant and if a bit is "1" then it is recessive. Thus, low priority numbers represent higher priorities. When a node with a recessive bit detects a dominant bit, then it knows it has lost the arbitration.

If priorities are unique, a MAC protocol using this scheme is not only prioritized, but is also collision-free. Hence, it is possible to schedule the bus such that if message characteristics (minimum inter-arrival times, transmission times, jitter, etc.) are known, then it is possible to compute upper bounds on message delays [99, 100, 101].

A relevant previous attempt was made to migrate the dominance protocol to the wireless context [102]. In that work, the approach considered was able to aperate in MBDs, but only offered a partial solution. A sending node transmits a busy tone on a separate channel and this tone has higher transmission power (or the receivers for the tone are more sensitive) so it has double the range as compared to the range of data transmission. This does not work in the case where two source nodes request to transmit to a receiving node and the two source nodes are close to each other but a communication obstacle keeps them hidden from each other (this problem is discussed in [79]).

Dominance/binary countdown protocols [4] are the main inspiration for the protocol proposed in Chapters 3 and 4.

2.4 Conclusions

This chapter provided background material on data aggregation and medium access control. A brief introduction to data aggregation is made and previous work in this area is reviewed. By this review, we conclude that previous data aggregation schemes

discussed have an execution time that depends on the number of nodes. In particular, no existing approach is able to perform efficient data aggregation in dense sensor systems where a single broadcast domain may contain a large number of nodes. This is a key motivation for the research carried out on this thesis.

This chapter also discussed several aspects related to MAC protocol design in wireless media, and overviewed the relevant literature on the subject, focusing on work were timing issues of the MAC behavior are considered.

Binary/Countdown protocols is an important family of MAC protocols for our research context: (i) it offers good properties in terms of providing timeliness support for event-triggered message models and (ii) it allows simultaneous non-destructive transmission of information in the same broadcast domain. These are key features for the challenges to be tackled in this thesis.

Chapter 3

WiDom for Single Broadcast Domains

Contents

3.1 Introduction

In this chapter, an adaptation of dominance/binary countdown MAC protocols to a wireless channel is proposed and validated. A relevant feature of this novel MAC protocol is that it can be exploited to efficiently obtain aggregate quantities in large scale dense networks, as discussed later in Chapter 5.

Achieving dominance in the wireless domain is challenging. To begin with, it is not possible to directly translate the behavior of wired protocols, as these require that nodes are able to transmit and receive at the same time. This is not possible in common radio transceivers, because the transmitted energy is much higher than the received energy. For this reason, dominance in wireless systems is achieved using a simple principle: when the transmitted bit is dominant, a pulse of a carrier wave is transmitted and there is no need to sense the medium. Conversely, when the bit to transmit is recessive, nothing has to be effectively sent, instead only the medium state has to be sensed.

Firstly, the protocol design and rationale behind it is proposed and discussed. An implementation of the proposed dominance protocol is presented and evaluated. This implementation is named WiDom. This MAC protocol also enables schedulability analysis over wireless networks. Thus, a response-time analysis for WiDom is developed and tested as well.

It is important to note that the solutions proposed in this chapter assume a Single Broadcast Domain (SBD) network. The extension of the approach to consider MBD will be carried out in Chapter 4.

This chapter also introduces the assumptions and notation used in this dissertation. These will be employed throughout Chapters 3, 4, 5.

3.2 Assumptions and Notation

Consider m computer nodes $N_1, N_2, ..., N_m$, each equipped with one radio transceiver. The system includes also n message streams $1, 2, 3, ..., n$, where a message stream is a continuous flow of recurrent message requests. A message stream is assigned to one node only, and nodes can have several message streams assigned to them. Let MAXNNODES denote an upper bound on m. It is assumed that MAXNNODES is known by the designer before run-time.

Workload. The exact time of a transmission request is unknown, but a lower bound on the time between two consecutive transmission requests from the same message stream is known. This lower bound is denoted as T_i. Every message from i requires C_i contiguous time units to transmit. The maximum time elapsed from the time instant of a request from i to the completion of the transmission of that message is called the response time of i, and it is denoted as R_i. Every time a message from i is requested to be transmitted it needs to finish the transmission at most D_i (the relative deadline of i) time units after it was requested.

It is assumed that when a node receives a message it does not send an acknowledgement. This assumption could easily be removed for unicast, by adding the acknowledgement time to the message transmission time.

Priorities. Priorities are assigned univocally to message streams; these priorities are non-negative integers in the range $0..2^{npriobits} - 1$, where $npriobits$ is the number of bits required to represent the priorities. This priority is denoted as an array of bits `prio[1..npriobits]`, where the most significant bit is `prio[1]`.

Propagation. The time-of-flight between two arbitrary nodes in the same broadcast domain is unknown, but it is non-negative and there is an upper bound α on the time-of-flights.

Clocks. Nodes are equipped with real-time clocks. These are not synchronized; that is, their values may be different. We consider that for every unit of real-time, the clock increases by an amount in the range $[1 - \varepsilon, 1 + \varepsilon], 0 < \varepsilon < 1$. Let CLK denote

the granularity of the clock. Performing state transitions on the protocol automaton takes a non-zero amount of time, and L represents the delay due to executing on a finite-speed processor.

Radio Transceivers. Unless explicitly stated (for example, when we develop and describe specific hardware to do so), nodes cannot send or receive out-of-band signals. Nodes use the radio transceiver to transmit messages and to send unmodulated pulses of a carrier wave. A node can sense other transmissions only if it is not transmitting.

The radio transceivers are characterized by three relevant timing parameters: T_{RX}, T_{TX} and T_{CS}. The transceivers take T_{RX} time units to switch from idle mode to reception mode and T_{TX} time units to switch from idle mode to transmission mode. We let T_{RXTX} denote $\max\{T_{RX}, T_{TX}\}$. T_{CS} denotes the time to detect a carrier wave when in receive mode.

Communication Links. Communication links are assumed to be bidirectional and the topology static while nodes are trying to access the medium. A data transmission that overlaps in time at a receiver causes a collision, and reception fails on that receiver. When one or more nodes transmit a carrier pulse at the same time, any listening node within range is able to detect the transmission of the carrier wave, as it is possible to detect the energy transmitted, independently of possible collisions. The communication range (R_{co}) is the maximum range at which two nodes N_i and N_j can communicate reliably. The carrier sensing range (R_{cs}) is the maximum range at which N_i can detect a transmission from N_j. The interference range (R_{it}) is the maximum range between nodes N_j and N_k such that simultaneous transmissions to N_j will collide with N_k. We assume that $R_{co} \leq R_{it} \leq R_{cs}$. This assumption is supported experiments reported in previous research [103].

Several different timeout values are used to describe the protocol. These timeouts are the constants that describe the platform (such as L, CLK, T_{RX}, T_{TX} or T_{CS}) and are also some protocol specific timeouts that are derived from the protocol automaton and the platform characteristics. These parameters are E, F, G, H, ETG. Section 3.3.2 discusses how to assign their values for the protocol designed for SBD networks and

Table 3.1: WiDom Parameters.

Parameter	Description
$npriobits$	Number of priority bits
α	Upper bound on the time-of-flight
L	Time for the execution of the protocol
CLK	Granularity of the clock
ε	Clock error
T_{RX}	Time to switch to reception mode
T_{TX}	Time to switch to transmission mode
T_{RXTX}	Maximum between T_{RX} and T_{TX} ($\max\{T_{RX}, T_{TX}\}$)
T_{CS}	Time to detect that a carrier wave is being transmitted
C_i	Time to transmit a message from message stream i
C	Time to transmit a message with the maximum payload allowed ($max_{i \in 1..n}\{C_i\}$)
MAX_TC	Number of tournaments after which a transition that forces the protocol to try to reset its state, for error recovery purposes
F	Initial idle period
E	Timeout to cope with synchronization imperfections (such as clock inaccuracies and transmit/receive switching times)
H	Duration of a priority bit
G	Guarding time interval between bits
ETG	Guarding time interval at the end of the tournament

Section 4.3.3 shows how to assign their values for the MDB case. Table 3.1 summarizes all protocol parameters and their meaning.

Notation of the Protocol Description

The protocol will be described using timed-automata like notation. States are represented as vertices and transitions are represented as edges. An edge is described by its guard (a condition which has to be true in order to make the transition) and an update (an action that occurs when the transition is made). In the automata, "/" separates the guards and the updates; the guards are before "/" and the update is after. The symbol "=" denotes test for equality and ":=" denotes assignment to a variable. When a timeout transition is enabled, it occurs immediately. The corresponding

update of that transition and a continuing path of enabled transitions occur at most L time units later. Note that the actual behavior is slightly different due to clock imperfection, time-of-flight of the carrier-signal and delays in the transitions. States are numbered and State 0 is the initial state. Associated to each node the following variables are considered: a clock x; an integer i within the range $0..npriobits - 1$; an integer prio occupying $npriobits$ bits; an integer $winner\_prio$ occupying $npriobits$ bits and a boolean variable `WINNER`. Let $winner\_prio[i]$ denote the bit i in the variable $winner\_prio$, and analogously for $prio[i]$.

Seven functions can be called by the MAC protocol in a node: `initRadio()`; `setRadioDataRxMode()`; `setRadioDataTxMode()`; `carrierOn()`; `carrierOff()`; `setCarrierSenseOn()`; `setCarrierSenseOff()` and `dequeueHPMsg()`. The call `initRadio()` is used to perform any initialization on the radio chip and to set it into a known starting state. The function `setRadioDataRxMode()` prepares the radio to receive a data packet, while the `setRadioDataTxMode()` sets the radio to packet transmission mode. The function `carrierOn()` starts transmitting a carrier wave and continues doing so until `carrierOff()` is called. Function `setCarrierSenseOn()` is used to set the radio into receive mode and start detecting carrier pulses, while the function `setCarrierSenseOff()` is called to stop detecting carrier pulses. To get the highest-priority message from the local queue of message requests, a node calls `dequeueHPMsg()`. The symbol "`carrier?`" is used with the following meaning: sense for a carrier and if there is a carrier then "`carrier?`" is true.

3.3 Design Aspects of WiDom-SBD

First, this section will briefly address the key aspects to be considered in the design of a correct dominance protocol for wireless networks in a SBD. Details are provided in Section 3.3.1 while the rational and correctness of the protocol are given in Section 3.3.2. Finally, a message's response time analysis is proposed in Section 3.3.3.

Basic Design Aspects

Synchronization

In dominance/binary countdown protocols, before nodes perform the collision resolution, they must agree on a time where this collision resolution starts, such that all nodes that want to transmit perform it at the same time.

This can be achieved by letting nodes wait for a "long" period of silence. This is similar to dominance/binary countdown protocols, and guarantees that nodes do not disturb an ongoing medium access resolution phase. After detecting this period of silence, a node may signal the start of the arbitration by sending a carrier pulse.

Tournament

In the protocol proposed in this section, when messages contend for the channel, a conflict resolution phase, similar to the dominance/binary countdown arbitration, is performed. In our protocol, this conflict resolution phase is named *tournament*. During the tournament, nodes transmit the priority of the message contending for the medium bit-by-bit. But, wireless transceivers can hardly be transmitting and receiving at the same time. Thus, when the transmitted bit is dominant there is no need to sense the medium, whereas, when the bit to transmit is recessive, nothing has to be effectively sent, instead only the medium state has to be sensed.

In this protocol, a bit of the tournament is different from a data bit. Each bit in the tournament has a fixed duration of time, which is considerably longer than a data bit. But, when a node wins the access to the medium, it may transmit at the full bit rate allowed by the specific radio transceiver.

3.3.1 Details of the Protocol

The protocol is formally presented in Figure 3.1, using timed-automata like notation. Figure 3.1 depicts the three main phases of the protocol: *synchronization*, *tournament*

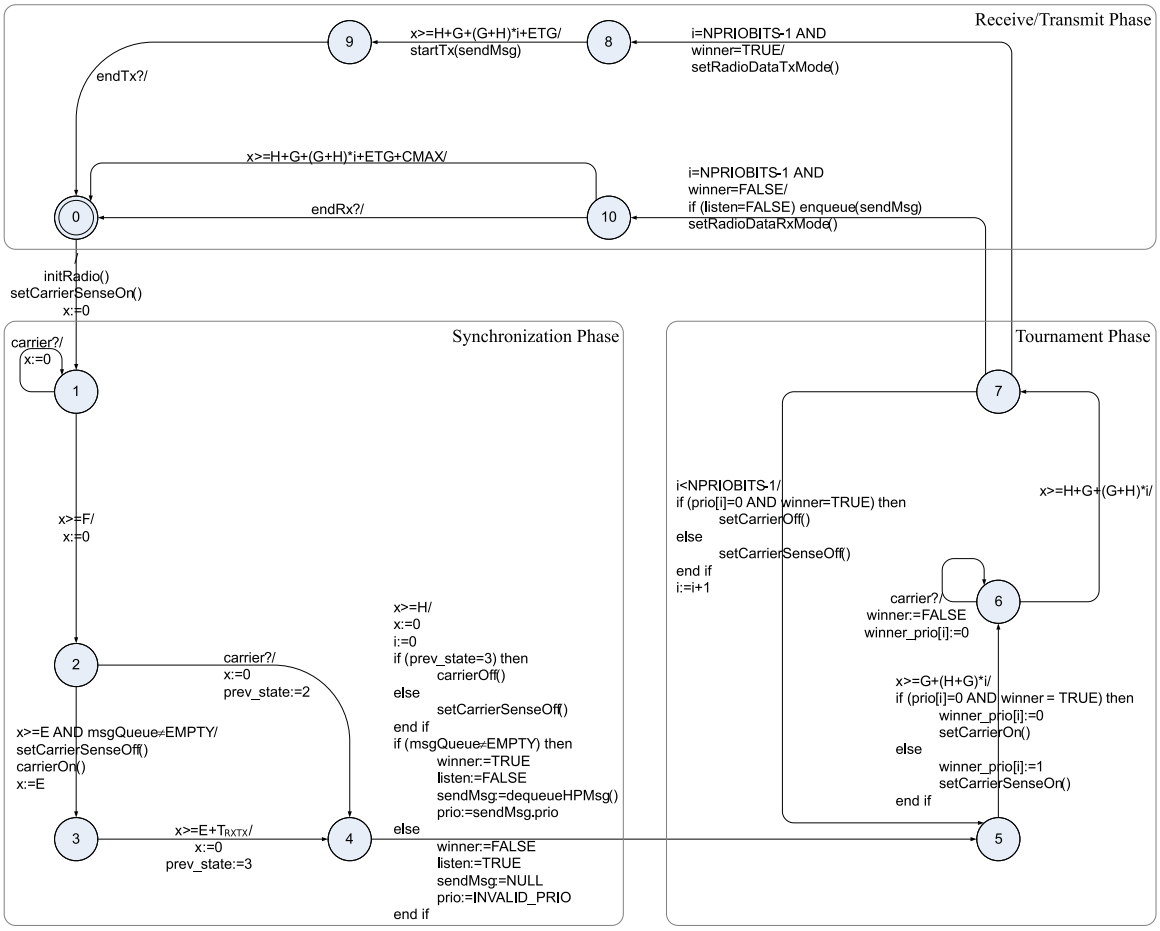


Figure 3.1: Details of the WiDom Protocol.

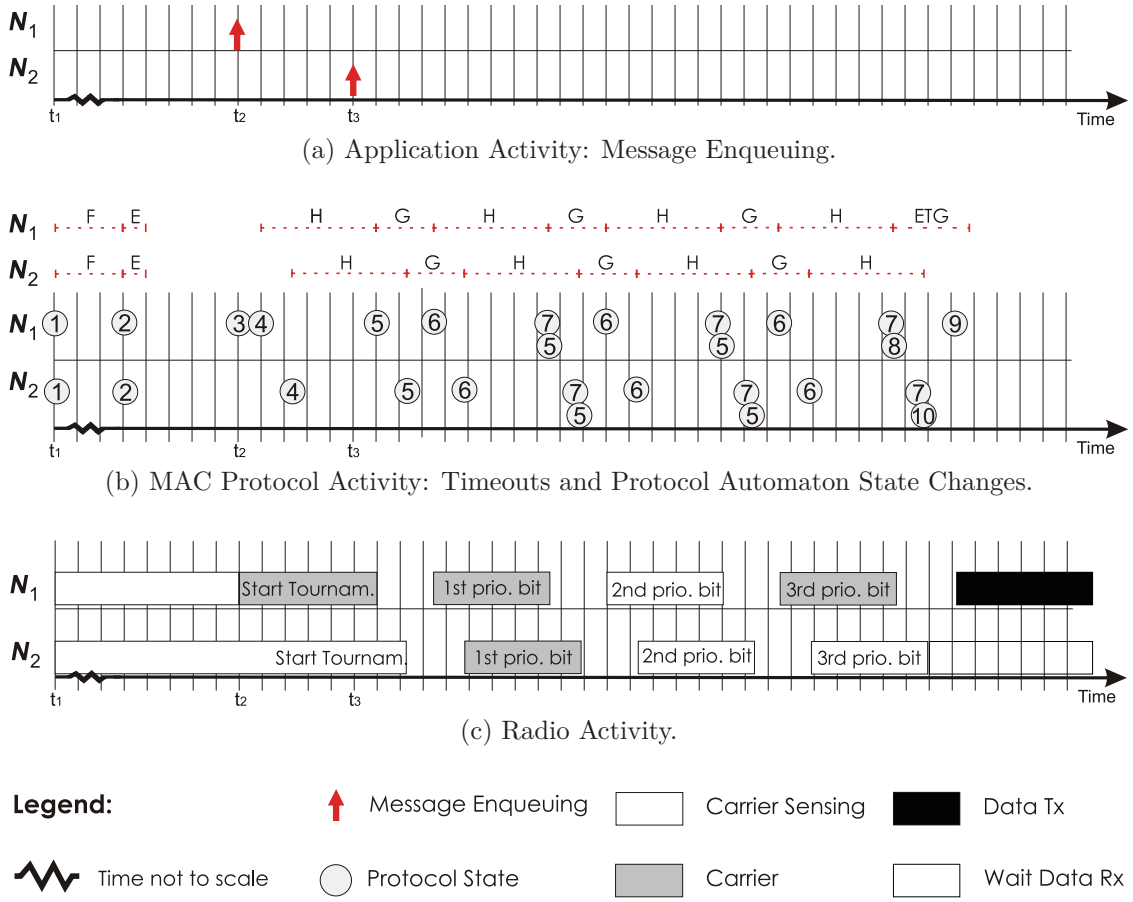
and *receive/transmit* phases. Nodes have to agree on a common reference point in time. This phase is called synchronization and happens before every collision resolution phase – the *tournament*. After the tournament, where nodes transmit the priority of the message contending for the medium bit-by-bit, nodes enter into the receive/transmit phase. In the receive/transmit phase, the node (assuming priorities are unique, there will be only one) that won the tournament proceeds to transmit its message and nodes that have lost will wait for the message reception or timeout.

States 1-4 in Figure 3.1 establish a common reference point in time between all nodes that request to transmit. In State 1, nodes wait for a period of silence (F) such that no node disrupts an ongoing tournament. Then, nodes with a pending message

perform transition $2 \rightarrow 3$ after E time units. This design is such that the duration of E encompasses possible clock differences between the nodes and guarantees that all nodes have time to listen for F time units of silence. Nodes that take $2 \rightarrow 3$ start sending a carrier pulse that signals the start of a tournament and establishes a common time reference. Other nodes may take one of the two following sequence of state transitions: (i) a node is in State 2 with pending messages and it did not hear a carrier for E time units, and so it makes the transition $2 \rightarrow 3$; or (ii) a node in State 2 (either because it is waiting to make transition $2 \rightarrow 3$, or it does not have any pending messages) detects the carrier pulse being sent by other nodes and performs transition $2 \rightarrow 4$. Nodes making transition $2 \rightarrow 4$ reset their timers. However, nodes making transition $2 \rightarrow 3$ wait T_{RXTX} time units to reset their timers because only at that time the carrier pulse is actually transmitted. And then stay in State 4 sending the synchronization carrier. Once in State 4, nodes make transition $4 \rightarrow 5$ after H time units. At this point, the synchronization ends with nodes resetting their timers.

The States 5-7 relate to the actual tournament. During the tournament, if a node loses the contention of a bit, then it will only proceed listening to find out which priority (also message identifier) wins the tournament. If a node does not lose the contention during this bit, it continues with the contention for the next bit. If the node contends with a dominant bit ("0") then it starts transmitting a pulse of the carrier in transition $5 \rightarrow 6$. If the node contends with a recessive bit ("1"), then in transition $5 \rightarrow 6$ the node starts performing carrier sensing. While at State 6, if a node contended with a recessive bit ("1") but heard a carrier wave, it has lost. After the tournament, the winning node (and there is only one winner of the tournament) makes the transition to State 8, waits for a while so that the radios of the other nodes can go into receive mode and then, at State 9, transmits the data part of the message. Then, it goes back to State 0.

Consider now a node which has lost the tournament. The node continues in the tournament and if it has a recessive bit, then it acts in the same way as if it had not lost. The reason for this is that with a recessive bit it just listens; it does not transmit



(a) Application Activity: Message Enqueuing.

(b) MAC Protocol Activity: Timeouts and Protocol Automaton State Changes.

(c) Radio Activity.

Figure 3.2: Activity Example in two Nodes (N_1 and N_2): Application (a), MAC protocol (b) and Radio (c).

a carrier wave. However, if a node has a dominant bit and it has lost (the boolean variable `WINNER` is `FALSE`), then the protocol acts differently from the case when it had won; no carrier wave is transmitted. After the end of the tournament, the node goes to State 10 waiting to receive the message or timeout. A node only receiving acts like a node losing the tournament from the start because the variable `WINNER` is assigned `FALSE` before the tournament (transition $4 \rightarrow 5$).

In order to understand the timeout parameters F, G, H, ETG and E, let us consider the activity of N_1 in Figure 3.2b. N_1 enters State 1 (denoted in Figure 3.2b with the symbol ①) at time t_1. From this time instant on, node N_1 starts monitoring the medium until it detects the initial idle time period, denoted by F. Every time

N_1 sends or tries to detect a carrier, it does so for H time units, representing the duration of a pulse of the carrier wave. The "guarding" time interval to separate pulses of carrier waves is denoted by G. This "guarding" time interval makes the protocol robust against clock inaccuracies, and takes into account that signals need a non-zero time to propagate from one node to another. ETG is the gap that a winner must introduce at the end of the tournament. Finally, E is a timeout used to improve the reliability introduced by imperfections imposed by the hardware during the synchronization (such as clock inaccuracies and transmit/receive switching times).

3.3.2 Rationale of the Design and Correctness

In this section we discuss the correctness of the protocol and demonstrate how assigning values to the constants E, F, G, H, ETG, T_{CS} and T_{RXTX} affect the correctness. The protocol must satisfy the following relevant properties (P3.1-P3.3):

Property 3.1 Mutual Exclusion*. At any given time, at most one computer node can be in State 9.*

Property 3.2 Progress*. There are two types of progress: (i) if a computer node is backlogged then State 0 is reached after at most C_i'' time units from any state; and (ii) if a message finishes transmission and there exists a backlogged node then one message of the backlogged nodes should be transmitted next.*

Property 3.3 Prioritization*. Of all nodes which were backlogged, the one that will transmit a message is the one that dequeues (at transition $4 \rightarrow 5$) the message with the highest priority.*

These properties hold if the following constraints (3.1-3.5) are satisfied.

Constraint 3.1 *When a node transmits a dominant bit in iteration i in the tournament, it is received by all other nodes and it is perceived to be received in iteration i.*

Consider an iteration of the tournament. It must have been sufficient overlap between the time where one node transmits the carrier to inform that it has a dominant bit and the time interval where a node with a recessive bit listens for nodes with a dominant bit. Due to clock drift and inaccuracy of synchronization, this overlap becomes smaller and smaller with the iterations within the tournament. Hence, the last iteration (the worst-case scenario) of the tournament is considered. Therefore, we derive the following inequality:

$$\begin{aligned}(H+G+(H+G)\times(npriobits-2))\times(1-\varepsilon)\\ -(G+(H+G)\times(npriobits-2))\times(1+\varepsilon)\\ -2CLK-L-2\alpha-T_{RXTX}-E>T_{CS}\end{aligned} \tag{3.1}$$

Inequality (3.1) guarantees that even in the presence of worst-case clock inaccuracies, all nodes will hear a dominant bit for at least the time necessary to detect a carrier (T_{CS}).

□

Constraint 3.2 *If a node N_i has perceived silence long enough (F time units) to make transition from State 1 to State 2 but other nodes perceive the duration of silence to be less than F, due to different time-of-flights and clock-imperfections, then node N_i needs to wait until all nodes detected this long time of silence.*

If inequality (3.2) is true

$$2CLK+L+2\alpha+F+2\varepsilon+T_{RXTX}<E \tag{3.2}$$

then the protocol must stay in State 2 for E time units, and this ensures Constraint 3.2 is true.

□

Constraint 3.3 *With similar reasoning as for (3.2), a node which has won the tournament must wait ETG time units before transmission (this occurs in $8 \rightarrow 9$) to ensure*

that all losing nodes reached State 10.

ETG must satisfy the following:

$$\begin{aligned} 2CLK + L + 2\alpha + (H + G + (G + H) \times (npriobits - 1)) \times 2\varepsilon \\ +(T_{RXTX} + E) < ETG \end{aligned} \tag{3.3}$$

□

Constraint 3.4 *During the tournament, the maximum time interval of idle time should be less than F, the initial idle period.*

This assures that if one node makes the transition from State 1 to State 2 (the initial idle time period) then all nodes will do it at most E time units later. Therefore, we have the following inequality:

$$\begin{aligned} [H + G + (H + G) \times (npriobits - 1) + ETG] \times [1 - \varepsilon] \\ - [H + G] \times [1 + \varepsilon] + 2CLK + L + 2\alpha < F \end{aligned} \tag{3.4}$$

□

Constraint 3.5 *The time interval between two successive dominant bits must assure that bits are interpreted correctly.*

The worst-case scenario occurs when these two bits are the last ones in the tournament. Therefore, the following inequality must also be satisfied:

$$\begin{aligned} [H + 2G + (H + G) \times (npriobits - 2)] \times [1 - \varepsilon] \\ - [H + G + (H + G) \times (npriobits - 2)] \times [1 + \varepsilon] \\ -2CLK + L + 2\alpha < F \end{aligned} \tag{3.5}$$

□

The values of E, F, G, H and ETG must be selected such that they satisfy inequalities (3.1)-(3.5). The selection of T_{CS} and T_{RXTX} is imposed by the platform chosen. Section 3.4.4 instantiates these timeouts for a concrete platform.

3.3.3 Response Time Calculations

Let us now introduce the messages' response-time calculations for the WiDom protocol. This analysis is based on the analysis for the CAN bus [101], which in turn builds upon existing analysis for non-preemptive static-priority scheduling [104]. But subtleties about the synchronization in the protocol require our schedulability analysis to deal with aspects that are not dealt with in [101].

Recall the sporadic model [105] with a system of n message streams: $1, 2, ..., n$. Each message stream i is characterized by T_i, D_i and C_i, as defined earlier in Section 3.2.

Let us first define the transmission time of a message considering the overhead of the protocol. We can do this by observing the automaton in Figure 3.1 again, traverse the path of the transitions of the winning node and observe the last timeout (the transition $8 \rightarrow 9$). Considering this path, we can define the time to perform the tournament when nodes are already synchronized and transmit a message as C_i':

$$\begin{aligned} C_i' = C_i + 2H + G + (G + H) \times (npriobits - 1) \\ + ETG + E + \max\{T_{CS}, T_{RXTX}\} \end{aligned} \tag{3.6}$$

where C_i is the time required to transmit a message from message stream i. The time to transmit a message and perform the tournament when nodes are not yet synchronized is denoted C_i'', and takes into account the initial idle time (F):

$$C_i'' = C_i' + F \tag{3.7}$$

Using the same reasoning as above, we can define $Q_{trnmt-SBD}$ as the time to perform a tournament considering the overhead of the protocol when nodes are not synchro-

nized:

$$\begin{aligned} Q_{trnmt-SBD} = 2H + G + (G + H) \times (npriobits - 1) \\ + ETG + E + \max\{T_{CS}, T_{RXTX}\} + F \end{aligned} \tag{3.8}$$

This will be the amount of time we want to minimize when trying to find the protocol parameters.

We will now address the formulation of the response time calculations assuming that the release jitter is zero. Release jitter has been addressed previously [106, 99], and this analysis can be adapted to consider such effect. The granularity of the time is $Q_{bit} = CLK$. Using these assumptions, the response time is (following a similar reasoning to [104]) as follows:

$$R_i = max_{q=0..Q}\{w_{i,q} + C''_i - q \times T_i\} \tag{3.9}$$

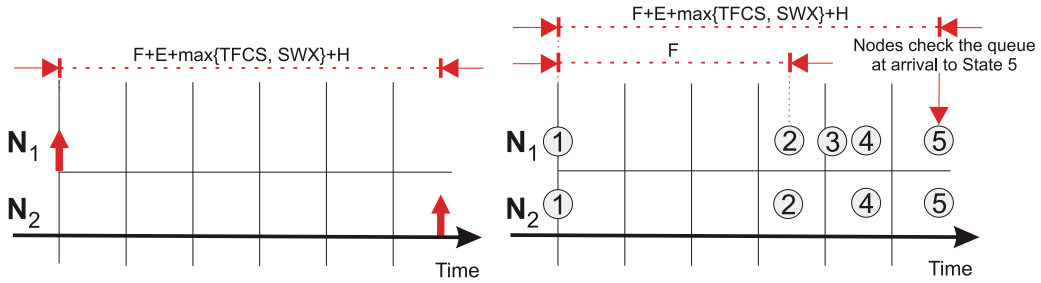
where $Q = \lfloor L_i/T_i \rfloor$. L_i is the length of the longest level-i busy period in non-preemptive context, which is given by the smallest positive integer L_i satisfying:

$$L_i = max_{j \in lp(i)}\left\{C'_j - Q_{bit}\right\} + \sum_{j \in hp(i) \cup i} \left\lceil \frac{L_i}{T_j} \right\rceil \times C''_j \tag{3.10}$$

where $hp(i)$ is the set of all message streams with priority higher than i, and, similarly, $lp(i)$ is the set of all message streams with priority lower than i. The waiting time $w_{i,q}$ is then:

$$\begin{aligned} w_{i,q} &= q \times C''_i \\ &+ \sum_{j \in hp(i)} \left(\left(1 + \left\lfloor \frac{w_{i,q} + Q_{TX} + Q_{bit}}{T_i} \right\rfloor \right) \times C''_j \right) \\ &+ max_{k \in lp(i)}\{C'_k - Q_{bit}\} \end{aligned} \tag{3.11}$$

Observe that Equation (3.11) differs from the analysis used in the CAN bus, as we have to add Q_{TX}, the time before the next message is dequeued after the previous



(a) Arrival Pattern of Messages. (b) Protocol Automaton State Changes.

Figure 3.3: Scenario that Maximizes the Response Time in WiDom.

message has been transmitted:

$$Q_{TX} = F + E + \max\{T_{CS}, T_{RXTX}\} + H \tag{3.12}$$

Figure 3.3 provides further intuition. Figure 3.3b shows two computer nodes N_1 and N_2 and how their states change as time progresses. Let us assume that there is a message stream 1 on N_1 and a message stream 2 on N_2 and that stream 1 has higher priority than stream 2. Figure 3.3a depicts the arrival pattern that maximizes the response time of stream 2. Observe that nodes check the queue of message transmission requests when they arrive to State 5.

The analysis considers the initial idle time between States 1-5 (Figure 3.1) to be part of the "message" when computing the interference. This initial idle period should not be included when computing the blocking. Thus the last term on the right hand side of Equation (3.11) uses C'_k instead of C''_k.

Let us apply the response time analysis to calculate the values according to Equations (3.9)-(3.11) in the following example (the response times will be tested experimentally in Section 3.4.4).

Example 3.1 *Consider $m = 10$ computer nodes with one message stream on each node. Message streams are given periods as shown in Table 3.2 (all values are in ms). Deadline monotonic is used to assign priorities, and $D_i = T_i$. We choose npriobits =* 10.

Table 3.2: Message Streams in Example 3.1.

i	1	2	3	4	5
T_i (ms)	200	400	800	1 600	3 200
R_i (ms)	82	134	186	291	343
i	6	7	8	9	10
T_i (ms)	6 400	12 800	25 600	51 200	102 400
R_i (ms)	395	552	604	709	731

It can be seen that the difference between the greatest T_i and the smallest T_i is very large; this is intentional because it is exactly for such workloads that prioritization is crucial in order to meet deadlines.

In order to apply the response time analysis, we need to instantiate the values of C_i' and C_i''. These values are a function of the protocol timeouts (CLK, T_{RXTX}, L, T_{CS}, E, F, G, ETG, H), which are platform dependant. The instantiation of these values is clarified in Section 3.4 (see Table 3.3). For the sake of this example, let us assume that all messages have the same C_i and applying Equation (3.6), yields $\forall i \in 1..n : C_i' = 29644\mu$s, and from Equation (3.7) it results that $\forall i \in 1..n : C_i'' = 52248\mu$s.

Applying Equations (3.9)-(3.11) results in the response times as shown in Table 3.2. Observe (in Table 3.2) that the scheduling theory predicts that all deadlines will be met because we obtain $\forall i : R_i \leq D_i$.

□

3.4 Implementation and Evaluation

There are a number of difficulties in implementing a wireless dominance protocol. There exist priority levels for which the protocol needs to switch between transmit and receive modes for every priority bit. Many transceivers are not designed for frequent switching, and hence every switching takes a non negligible amount of time. It is also well known that wireless channels typically have significantly higher noise levels than wired channels, and that detection of pulses of short duration is difficult [50].

Therefore, it is very relevant to demonstrate an effective implementation of dominance protocols for wireless medium. This section addresses the implementation of the proposed protocol in commercial-off-the-shelf (COTS) platforms.

3.4.1 Sensor Network Platforms

To demonstrate the feasibility of the protocol, several implementations in real-world platforms were carried out. These implementations are reported in the next section (Section 3.4.2) and in Section 4.4. Additionally, Chapter 6 presents another implementation of WiDom in the form of an addon board that can be plugged into the CMU-FireFly [80] and MicaZ [107] (and the whole family of Mica sensor network platforms).

The CMU-FireFly and MicaZ are two very similar sensor network platforms, offering a low power microcontroller (ATmega128(1) microcontroller), 128 kbytes of program flash memory and an IEEE 802.15.4 compliant radio transceiver CC2420 [108] capable of 250 kbps data rate.

The CMU-FireFly and MicaZ platforms are an attractive alternative for the implementation because of the following relevant characteristics: (i) they allows replacing the existing MAC protocol easily; (ii) the available timers are sufficiently precise; (iii) the radio can be put into a specific test mode, where it is possible to transmit a non-modulated carrier for an arbitrary duration; (iv) the radio has built-in receive signal strength indicator (RSSI)/energy detection functionality and clear channel assessment (CCA) is available through a digital output pin.

Dominance protocols in wired media require that a node can simultaneously transmit while it detects the transmissions from other nodes. Unfortunately, this is not possible in most radio transceivers, including the CC2420, because the transmitted energy is much higher than the received energy. For this reason, the CC2420 can only be either in transmission mode or in reception mode, and it can take up to 192 μs to switch between these two modes.

The CC2420 radio can be set into a transmitter test mode to either transmit a modulated carrier or a non-modulated carrier wave. The RSSI obtained with a non-modulated carrier is 9dBm stronger than the one obtained when transmitting a modulated carrier [108]. Hence, the non-modulated carrier is used for transmitting the carrier waves during the tournament. It is also necessary to detect when other nodes transmit a carrier wave. For this, the CC2420 support for CCA is used. The CCA functionality of the CC2420 radio computes the average RSSI over the last 128 μs. This average is compared to a configurable threshold and then CC2420 sets the CCA output pin accordingly. This pin is sampled by our software communication stack to detect if other nodes are sending carrier pulses. Every time the radio is set into receive mode, it takes at least 128 μs to make the first valid CCA operation.

The proposed protocol is heavily dependent on timers. The Atmega128 microcontroller in the two platforms provides two 8-bit timer/counter and two 16-bit timers. The implementation needs one of these timers/counters, and the one actually used depends on the one being used by other software running on the platform.

3.4.2 Implementation Overview

The prototype implementation of WiDom for SBD was developed using TinyOS 1.x [109], an open-source operating system designed for wireless sensor networks. This implementation was done for the MicaZ [107] sensor network platform. The main TinyOS software components of the implementation are presented in Figure 3.4, which also provides a simplified overview of the implementation component assembly. Rectangles are implementation modules of components. For the sake of simplicity, only the most relevant modules and interfaces are depicted, and configurations wiring components are omitted. Shaded rectangles are TinyOS components reused in the implementation. Triangles pointing into a rectangle are provided interfaces. Triangles pointing out represent used interfaces. The names of the provided interfaces are in italics.

In Figure 3.4, components `CC2420Control`, `HPLCC2420FIFO`, `HPLCC2420Interrupt`

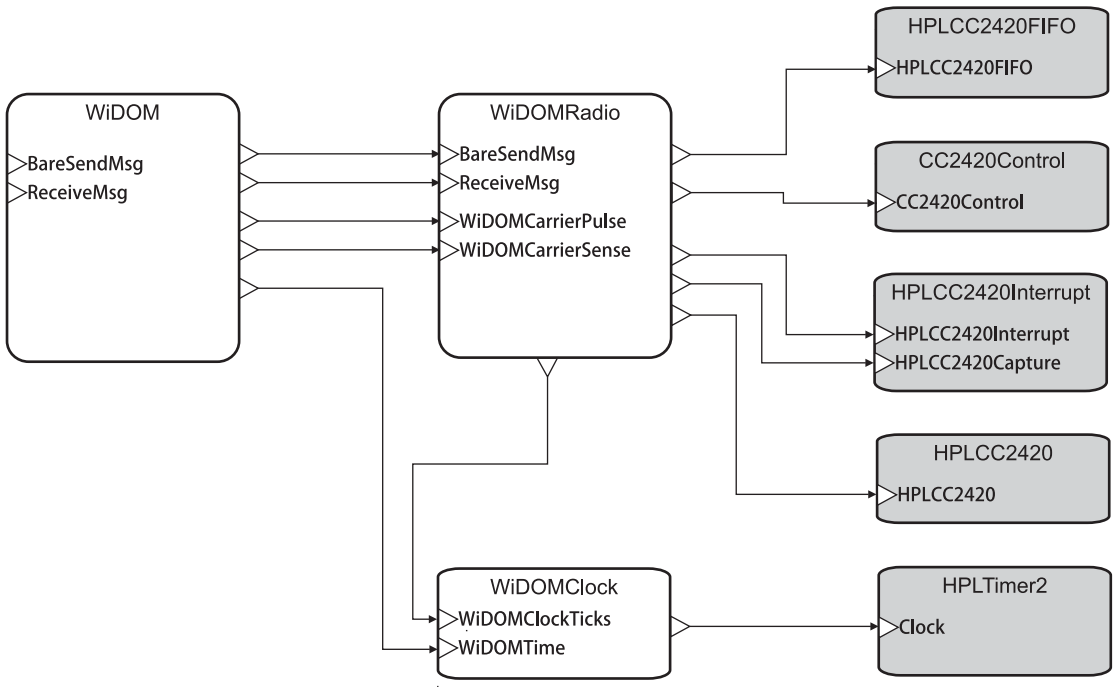


Figure 3.4: TinyOS Component Assembly for the WiDom Implementation.

and `HPLCC2420`, are part of the Hardware Presentation Layer (HPL) for the CC2420, and are regular TinyOS components reused by the implementation. These components provide basic functionalities for handling the radio.

The component `WiDomClock` uses `HPLTimer2` (also an HPL component of TinyOS) to drive the timing of the protocol. `WiDomClock` configures the timer prescaler to deliver the timer interrupts every 34.722 μs, which is used to drive the timing maintained by this component. For this reason, when timeouts are selected, these will be multiples of 34.722 μs.

The `WiDomRadio` component is responsible for providing all radio functionalities needed by the MAC protocol. It handles the interactions with the HPLs for sending/receiving packets, and it also provides the `WiDom` component with a simple send/receive interface (interfaces `BareSendMsg` and `ReceiveMsg` are commonly used TinyOS interfaces for these purposes). Finally, the `WiDom` component implements the WiDom protocol following closely the protocol's specification as depicted in Figure 3.1.

Table 3.3: WiDom-SBD Parameters for the COTS Experimental Platform.

Parameter	Value	Notes
$npriobits$	10	User/Platform-defined parameter
α	1 μs	Defined according to expected node geographical distribution
L	5 μs	Platform-defined parameter
CLK	34.722 μs	Platform-defined parameter
ε	10^{-5}	Platform-defined parameter
T_{RXTX}	347 μs	Platform-defined parameter
T_{CS}	486 μs	Platform-defined parameter
C_m	2 176 μs	Platform-defined parameter
F	22 604 μs	Computed from protocol constraints
E	416 μs	Computed from protocol constraints
H	1 458 μs	Computed from protocol constraints
G	972 μs	Computed from protocol constraints
ETG	798 μs	Computed from protocol constraints
$Q_{trnmt-SBD}$	50 062 μs	Computed from protocol constraints

3.4.3 Instantiating the Protocol Parameters.

With a concrete implementation of the protocol, we can now instantiate the time-out parameters of the protocol. These parameters must be selected according to the constraints given in Section 3.3.2 and the platform-dependent parameters.

The characteristics of the platform previously described lead to the concrete parameter values shown inTable 3.3. The time to transmit a message in the CC2420 is computed as follows. Assume that the data length of messages is 64 bytes (one byte for the length of data in a packet is included). Adding 3 bytes for the preamble and 1 byte for the start frame delimiter, the time to transmit a message is then given by:

$$C_i = ((64 + 3 + 1) \times 8) \times \tau_{bit} \tag{3.13}$$

where τ_{bit} is the transmission time for a single bit ($\tau_{bit} = 4\mu$s, assuming a bit rate of 250 kbits/s [108]).

The time to switch between to transmission mode (T_{RXTX}) is derived from the datasheet, taking into account the maximum time the radio takes to switch from

one mode to the other (192μs), added to the time until the first CCA operation (128μs). Since this parameter is measured in runtime by the timer on the platform, this value is then rounded up to the nearest clock tick. The value of T_{CS} was determined experimentally, as it will be described later in Section 3.4.4. The value of L was assigned to allow for about 40 instructions to be executed on the microcontroller for each state transition of the protocol.

A choice of parameters F, E, H, G and ETG that satisfies constraints (3.1-3.5) is also given in Table 3.3. These parameters were obtained simply by formulating the following optimization problem:

minimize $Q_{trnmt-SBD}$**, *subject to Constraints* 3.1-3.5 (in Section 3.3.2).**

$Q_{trnmt-SBD}$ is minimized because it reflects the time to perform a tournament considering the overhead of the protocol when nodes are not synchronized, and this is the amount of time we wish to minimize for all message streams. A linear programming solver was used to obtain a solution. Since these parameters are measured in runtime by the timer on the platform, the values obtained were rounded up to a multiple of the clock granularity.

3.4.4 Experimental Results

Determining T_{CS}. An initial experiment was testing the ability of our target platform to detect pulses when the receiver is always in receive mode. Two nodes were used, one sender and one receiver, with non-obstructed line-of-sight and with them 15 meters apart. The experiment was conducted in an indoor office environment, with the nodes located at the ends of a corridor. The default values as described in the manufacturer's manual for the radio parameters were used and the experiment was conducted with new batteries in the nodes. Different durations of the pulses were used, being their durations set as multiple of CLK. For every duration, 100 000 pulses were transmitted, the number of detected pulses was counted, and was computed the estimated probability of an undetected pulse. The result of this experiment is shown

Table 3.4: Probability of an Undetected Carrier.

Pulse Duration (μs)	Probability of Undetected Carrier (%)
138	54 %
277	46 %
486	0 %

in Table 3.4. As shown, for a duration of 277 μs and bellow the receiver does not detect the pulse, while for a pulse duration of 486 μs, all pulses were detected. Hence, we can conclude that the probability of detecting a carrier using pulses of this duration is very high.

Evaluation Hypotheses. For the experimental evaluation of the protocol, presented in the remainder of this subsection, we put forward the following hypotheses:

Hypothesis 3.1 *The implementation of the protocol offers collision-free medium access for data messages;*

Hypothesis 3.2 *The implementation of the protocol offers prioritized medium access;*

Hypothesis 3.3 *The response-time analysis equations in (3.9)-(3.11) can be used to analyze the response-times of the implementation of the protocol.*

Hypothesis 3.1 and Hypothesis 3.2. In order to test Hypothesis 3.1 and Hypothesis 3.2, four experiments were conducted. Let d denote the maximum distance between any two nodes. A set of m nodes were positioned in a circle such that for every node the distance to its neighbors with the minimum distance is maximized (this placement was selected only for the convenience of the experiment's description). The experiment runs as follows. A special node (which is not included in the m nodes) transmits a carrier wave and all other nodes boot. All nodes request to transmit a message and enter in State 1 (refer to Figure 3.1). These nodes stay in State 1 until the special node stops transmitting the carrier. The experiment was performed for $m = 2$ and $m = 10$. For the case of $m = 2$, all nodes make a new request to transmit

Table 3.5: Probabily of Correct (Collision-Free) Reception an Prioritization.

m	$d = 1\ meter$	$d = 4\ meters$
2	100.000%	100.000%
10	100.000%	99.998%

a message using `random`$(0, 255)$ ms (this means generating a uniformly distributed random number in the interval [0, 255]) after the previous message request. For the case $m = 10$, all nodes make a new request to transmit a message `random`$(0, 1023)$ms time units after the previous request. The diameter was also varied; one experiment had $d = 1\ meter$ and the other $d = 4\ meters$. Nodes are given IDs from 1 to 10 and their priority is equal to the ID. All messages have 64 bytes.

Every node has a sequence counter, initialized to 1. The sequence counter is transmitted in every message and then the sequence counter in the node is incremented. Whenever a node receives a message, it compares the received sequence counter to the sequence number previously received from the same node. The difference between the received sequence counter and the previously stored, allows checking if transmissions are collision-free. Since a collision leads to a lost message, this process gives an upper bound on the number of collisions.

Prioritization was also tested. This was done as follows. When a node sends a message it sends its priority in the data packet. All nodes receive this packet (if they do not receive, it would be considered as a collision, see Hypothesis 3.1) and if the priority of the winner is less than the priority of this node then it is considered as a prioritization error.

The results of the experiments are presented in Table 3.5. Each experiment was performed for a total of 100 000 messages; observe that more than 99.998% of all messages were collision-free and prioritized. This corroborates Hypotheses 3.1 and 3.2.
□

Table 3.6: Message Streams Used to Test Hypothesis 3.3.

i	1	2	3	4	5
T_i (μs)	256 000	512 000	1024 000	2 048 000	4 096 000
R_i (μs)	80 415	132 835	185 255	237 675	342 515
i	6	7	8	9	10
T_i (μs)	8 192 000	16 384 000	32 768 000	32 768 000	32 768 000
R_i (μs)	394 935	447 355	499 775	709 455	733 880

Table 3.7: WiDom-SBD Parameters Used in Section 3.4.4.

Parameter	Value	Notes
npriobits	10	User/Platform-defined parameter
α	1 μs	Defined according to expected node geographical distribution
L	5 μs	Platform-defined parameter
CLK	34.722 μs	Platform-defined parameter
ε	10^{-5}	Platform-defined parameter
T_{RXTX}	347 μs	Platform-defined parameter
T_{CS}	486 μs	Platform-defined parameter
C_m	2 176 μs	Platform-defined parameter
F	24 409 μs	Computed from protocol constraints
E	312 μs	Computed from protocol constraints
H	1 562 μs	Computed from protocol constraints
G	729 μs	Computed from protocol constraints
ETG	555 μs	Computed from protocol constraints
$Q_{trnmt-SBD}$	50 234 μs	Computed from protocol constraints

Hypothesis 3.3. In order to test Hypothesis 3.3, two experiments with the message set as described in Table 3.6 were conducted. Note that the values shown in Table 3.6 were computed considering the protocol parameters in Table 3.7 (in these experiments we did not use the parameters show in Table 3.3). All messages have the same C_i and applying Equation (3.6), yields $\forall\ i\ \in\ 1..n:\ C'_i\ =\ 28011\mu$s, and from Equation (3.7) it results that $\forall\ i\ \in\ 1..n:\ C''_i\ =\ 52420\mu$s.

The only difference between the two experiments performed was that message streams were strictly periodic in one experiment while, in the other, message streams were sporadic so that a node with a message stream i made a new transmission request T_i+`random`$(0.5 \times T_i)$ ms after the previous request. Every node had one message

Table 3.8: Response Times Observed (Periodic Message Streams).

i	1	2	3	4	5
$Minr_i$ (μs)	25 752	25 787	25 926	27 002	26 620
$Avgr_i$ (μs)	34 363	35 891	35 822	46 551	48 322
$Maxr_i$ (μs)	118 738	131 932	131 793	182 105	184 987
R_i (μs)	80 415	132 835	185 255	237 675	342 515
Deadline miss prob.	0.007%	0.000%	0.000%	0.000%	0.000%
i	6	7	8	9	10
Min (μs)	25 891	27 627	27 627	27 627	27 627
Avg (μs)	87 523	48 738	47 662	57 662	47 627
Max (μs)	187 071	181 029	184 328	261 063	179 536
R_i (μs)	394 935	447 355	499 775	709 455	733 880
Deadline miss prob.	0.000%	0.000%	0.000%	0.000%	0.000%

stream, thus $n = m$. The experimental setup was as follows. In each node, messages were requested to be sent periodically or sporadically, depending on the experiment. Nodes count the time since a message is requested to send until it is actually transmitted. This time (q_i, the measured queuing time of the message i) is sent in the data payload of the packet. A special node (not included in the m nodes) receives the messages, gets the queuing time from the data payload and calculates the response time (the response time calculated from the measurements is denoted as r_i). From the reasoning in Section 3.3.3, the response time is:

$$r_i = q_i + C_i \tag{3.14}$$

The experiments were run for the message streams as defined (the periods) in Table 3.6, until 100 000 messages were transmitted. The results obtained with this experiment are shown in Tables 3.8 and 3.9 (all values are given in μs). The upper bounds on the response times, R_i, computed using the equations from Section 3.3.3, are also presented.

From Tables 3.8 and 3.9 it is possible to observe that a small number of the measured response times were above the calculated upper bounds. The cause of the deadline misses was further investigated experimentally in our embedded computer plat-

Table 3.9: Response Times Observed (Sporadic Message Streams).

i	1	2	3	4	5
$Minr_i$ (μs)	25 752	25 752	25 891	26 065	27 627
$Avgr_i$ (μs)	33 079	36 829	38 460	43 252	45 509
$Maxr_i$ (μs)	97 210	168 807	184 814	218 980	240 334
R_i (μs)	80 415	132 835	185 255	237 675	342 515
Deadline miss prob.	0.014%	0.009%	0.000%	0.000%	0.000%
i	6	7	8	9	10
$Minr_i$ (μs)	27 627	27 627	27 627	27 627	27 627
$Avgr_i$ (μs)	47 349	47 349	48 565	44 954	44 190
$Maxr_i$ (μs)	240 056	240 056	224 258	214 848	235 786
R_i (μs)	394 935	447 355	499 775	709 455	733 880
Deadline miss prob.	0.000%	0.000%	0.000%	0.000%	0.000%

forms as follows. A lightweight logging mechanism was implemented in the nodes. Due to the small amount of memory, nodes only recorded the sequence of state transitions from the last tournament (with local timing information). The nodes were configured to check the queuing time of outgoing messages with the value computed as given analitically from Section 3.3.3. When a node detected a deadline miss, it would stop normal operation and output the state transitions of the last tournament via the serial port. With this, it was found that occasionally nodes perceived noise when waiting for F time units of silence (transition $1 \rightarrow 2$ in Figure 3.1), and this caused a deadline miss.

To make the protocol less prone to noise, the transmit power during the tournament could be increased and the sensitivity of receivers decreased in order ro make them less susceptible to noise. One could also enforce that a receiver must have detected a carrier pulse of a given duration in order to perceive it as a detected carrier, such that it is ensured that short spikes of noise are not mistaken as being a carrier. Other efforts in The improving the reliability of WiDom are discussed later in Section 6.4.

Nonetheless, note that WiDom is designed to perform well in the presence of noise: a node waiting for F units of idle channel may need to restart its waiting period due to noise, and noise can cause a node to perceive a dominant bit when there is only

recessive bits. However, these scenarios do not cause a collision. In particular, was noticed that if a node experiences strong noise for several seconds, then the protocol will simply lose its tournament and does not start any new ones during the duration of noise and then the noise is over, WiDom operates normally, attempting to send the messages that were never transmitted during the duration of the noise.

Note that, in a scenario where timeliness guarantees are a major concern, it is necessary to perform proper planning of radio frequencies and node positioning within the field in order to further improve the robustness of wireless communication. Such engineering work is common practice today, and software tools for supporting those efforts are available. Note, however, that the deadline miss probability provides evidence that the protocol performs effectively, and that this experiment corroborates Hypothesis 3.3.

□

3.5 Conclusions

In this chapter, a novel wireless MAC protocol – WiDom – was proposed. The inspiration of WiDom was dominance/binary countdown protocols, used long ago in wired networks such as CAN. Implementing binary countdown/dominance protocols in wireless is challenging and it was successfully tackled in this work.

The reason for endorsing this challenge is that dominance-based MAC protocols have important characteristics for the research objective of this work (efficient and scalable data processing in large-scale, dense sensor networks), notably: (i) it allows simultaneous non-destructive transmission of information when nodes share the same broadcast domain, and (ii) it offers good timeliness properties for handling sporadic message requests.

In this chapter, it was proved that WiDom is collision-free, does not require synchronized clocks and supports a large number of priority levels. Such a large number of priorities can be supported by other prioritized protocols (see e.g., [103]) only at

the cost of an overhead several orders of magnitude higher. The experimental evaluation of WiDom shows that the probability that a message is transmitted collision-free, correctly prioritized and received (neither lost nor corrupted) by all other nodes is high, and this reliability justifies the study of schedulability analysis techniques for sporadic messages in wireless networks; hence a response-time analysis for WiDom was developed and tested as well.

WiDom is specially suited to provide pre-run-time guarantees for sporadic message streams. This is also a very important feature of the protocol. Because emerging embedded systems are dealing with the physical world, it is an important requirement that their data services are able to meet timing constraints. Stimuli from the environment are typically triggered sporadically (here, sporadically means that the exact time of an event is unknown, but a lower bound on the time between two consecutive events of the same type can be defined), and the messages generated in such systems will generally follow a similar pattern. While many scheduling algorithms and analysis techniques for wireless communications are available for periodic messages, the case of sporadic messages is less studied. Most of the current wireless protocols cannot be analyzed to offer pre-run-time guarantees that sporadic messages meet deadlines, and the protocols that do offer such guarantees rely on polling, which is inefficient when the deadline is short and the minimum time between two consecutive requests is long.

A protocol providing an upper bound on the queuing times of messages is naturally useful for supporting scheduling of real-time traffic. For unicast communication, acknowledgements and retransmissions can be introduced without any modification to the protocol. The protocol, parameters and the response-time analysis presented are easily extendable to consider the time for a receiver to send an acknowledgement. Similarly, a technique like sending several replicas of the same message after the tournament can be introduced to improve reliability.

The other important characteristic of WiDom is that it can be efficiently exploited for performing scalable computations of certain aggregate quantities. This is a key aspect in this thesis and therefore Chapters 5 and 6 will be devoted to this effort.

Finally, some notes concerning improvements of the practical realization of WiDom.

As it can be perceived from both the proposed timed-automata and the experiments reported in this chapter, WiDom implies a significant additional overhead. Part of this overhead is due to technological limitations. The overhead of the proposed protocol can be minored with carefully chosen hardware and more sofisticated techniques on how to use it. This will be further addressed in Chapter 6.

The protocol presented in this chapter can be vulnerable to faults in detection the carrier, and these faults can lead to erroneous tournaments. In Chapter 6, it is described a simple technique to improve the reliability of the protocol.

Chapter 4

WiDom for Multiple Broadcast Domains

Contents

4.1 Introduction

The wireless version of the bit dominance MAC protocol presented in Chapter 3 was designed for a SBD. In the case of cooperative applications covering a relatively large geographical area, the assumption of having a SBD will not hold, and therefore there is the need to deal with a well-known phenomenon in wireless networks called the hidden node problem (already described in Section 2.3.2). Previous work within the wireless networking community offered MAC solutions to the hidden node problem, but they were either not prioritized or they depended on out-of-band signaling.

To overcome these limitations, WiDom will be extended for wireless networks where a broadcast from a node does not necessarily reach all other nodes in the network (that is, consider MBD), consequently the hidden node problem may be present. The proposed solution is the first prioritized and collision-free MAC protocol designed to successfully deal with hidden nodes without relying on out-of-band signaling. The protocol is evaluated experimentally both using simulation and real-world platforms to prove the protocol correctness.

4.2 Assumptions and Notation

In this chapter most of the same assumptions and notations introduced described in Section 3.2 will hold. In addition, the following definitions will be used for convenience of the protocol description.

Definition 1 Neighbor. *We say that a node N_i is a neighbor of node N_j if N_i is within R_{cs} range of N_j, for $\forall i, j \in 1..n : i \neq j$. As an example, in Figure 4.1, node N_4 is a neighbor of N_5 and N_3. N_6, for example, is not a neighbor of N_4.*

Definition 2 2-Neighbor. *We say that a node N_i is a 2-neighbor of node N_j if either: (i) N_i is a neighbor of N_j; or (ii) there exists a node N_k such that N_i is a neighbor of N_k and N_k is a neighbor of N_j. In Figure 4.1, nodes N_1 and N_3 are 2-neighbors. In addition, N_1 and N_2 are also 2-neighbors. Conversely, N_4 is not a 2-neighbor of N_1.*

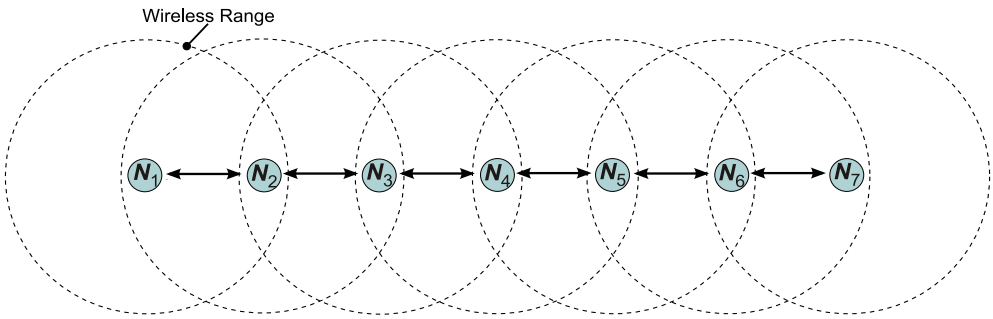


Figure 4.1: Example Topology Illustrating Neighbor and 2-Neighbor Nodes.

4.3 Design Aspects of WiDom-MBD

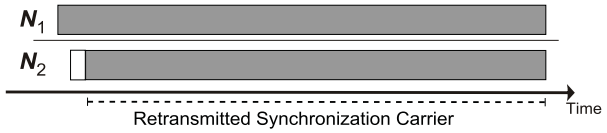
4.3.1 Overall Design Options

This section covers key aspects to be considered in the design of a correct dominance protocol for wireless networks with MBDs namely, synchonization and tournament issues.

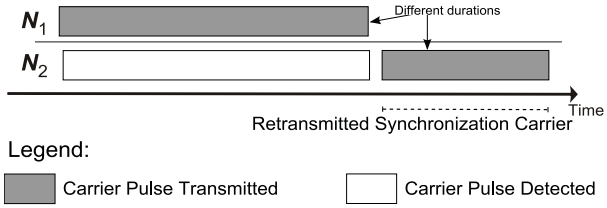
Synchronization

Prior to each tournament, nodes need to perform a synchronization phase where they agree on a common time reference. This synchronization is essential so that nodes correctly perform the tournament.

A node must be synchronized with at least its 2-neighbors. For example in Figure 4.1, assume that N_4 requests to transmit the highest priority message in the overall system. In order for N_3 and N_5 to correctly receive the message from N_4, it is necessary that not only nodes in direct range (N_3 and N_5) of N_4 refrain from transmitting, but also that N_2 and N_6 do not transmit as well (observe that nodes N_2, N_3, N_5 and N_6 are 2-neighbors of N_4). Therefore, a tournament must involve the set of 2-neighbor nodes. In order to allow message arbitration, all 2-neighbor nodes must be synchronized (a requirement for a correct tournament) and priority bits must be propagated to all 2-neighbors during the tournament. In this case, N_1 and N_7 do not cause any interference to data transmissions from N_4, because N_1, N_7 and N_4 do not share any



(a) Immediate Retransmission of Synchronization Carrier.



(b) Using Different Durations in the Synchronization Carrier.

Figure 4.2: Alternatives to Retransmit the Synchronization Carrier.

direct receivers. Propagating priority bits more than two hops away would prevent N_1 and N_7 from transmitting a message in parallel with the message from N_4.

The remaining of this section discusses how to achieve 2-neighbor wide synchronization across the network without requiring global time synchronization.

For the case of a single broadcast domain (refer to Chapter 3), synchronization is achieved by letting a node wait for a "long" period of silence and then sending a carrier pulse. The new protocol uses a similar approach. A node that wishes to transmit monitors the medium for a "long" period of silence. After this silence period, the node starts sending a carrier pulse. This carrier pulse signals that the node will start a tournament while also establishing a time reference with other listening nodes. This carrier pulse is called the *synchronization carrier pulse*.

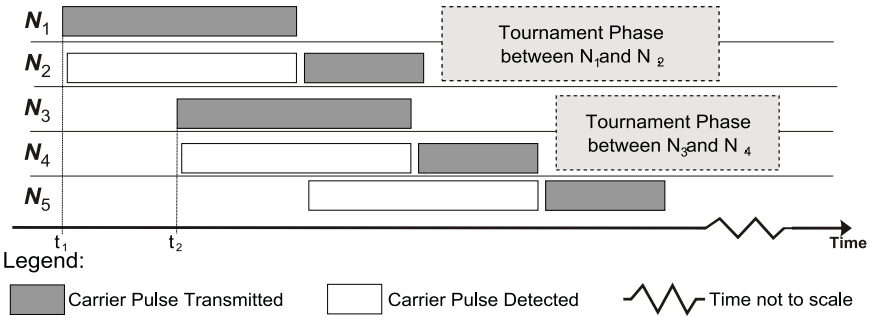
In order to provide two hop synchronization, the carrier must be retransmitted. Any node that detects a synchronization carrier being transmitted will immediately start transmitting its own synchronization carrier. This is illustrated in Figure 4.2a for two nodes N_1 and N_2. This solution causes the synchronization carrier pulse to be propagated network wide. To avoid this, one could try to differentiate between the carriers that are directly transmitted from a node within radio range and those that are retransmitted carriers. Nodes could only retransmit carriers from a node within a single hop. The only way to do this (without out-of-band signaling) is to either detect

a different duration of the carrier used for synchronization, or use different carrier patterns (predefined combinations of pulses and silence). Figure 4.2b depicts the former solution. In this case, nodes cannot start retransmitting the carrier immediately. They need to wait until they are able to decide whether they are receiving a transmission or a retransmission of a synchronization carrier. The latter solution is similar, but a node sends a "synchronization carrier" by sending a combination of carrier pulses and silence. A node monitoring the medium will only detect a "synchronization carrier" if it observes this pattern.

Unfortunately, these techniques (depicted in Figure 4.2) cannot be applied to bound the synchronization among 2-neighbors. As depicted in Figure 4.3, two or more nodes that start synchronization at different points in time may mislead a third node. Nodes N_1 and N_3 are more than two hops away from each other. Assuming they start sending synchronization pulses at time t_1 and t_2, respectively, this will cause the nodes one hop away (N_2 and N_4) to retransmit the synchronization carrier (note that the carrier pulse retransmitted has a smaller duration). The problem arises when a node N_5, two hops way from both N_1 and N_3 detects the retransmission of these two synchronization carrier pulses and takes them as a synchronization pulse sent by a node only one hop away. This will drive N_5 to start retransmitting the synchronization carrier, disrupting the tournament already started by nodes N_1, N_2, N_3 and N_4. A similar problem occurs when using different patterns for the synchronization carriers. This is illustrated in Figure 4.3b.

Immediately retransmitting the synchronization carrier arbitrarily far away (as depicted in Figure 4.2a) may appear to drastically impact performance. If we reason on a topology arbitrarily large (i.e. having a large number of broadcast domains), then we can see that this is not true; the entire network is not silenced for each transmission. First, although synchronization pulses must be propagated throughout the entire network, it is still possible for many nodes to transmit data messages in parallel (as already mentioned). While the synchronization wave is transmitted, another node that has not received the pulse yet can initiate its own tournament. Since carrier pulses do

(a) Network Topology for Scenario in Figure 4.3b.



(b) Unsynchronized Retransmission Problem.

Figure 4.3: Synchronization Carrier Retransmission Problem.

not collide like normal data packets, the multiple carrier waves will simply merge into each other. Second, the duration of a priority bit is affected by the synchronization error among 2-neighbors but is independent of the synchronization error between any two nodes in the network more than two hops way from each other, and hence it is independent of the network diameter.

This scheme also guarantees progress as all nodes will either start a tournament themselves (thus sending a synchronization pulse) or detect and retransmit a synchronization pulse.

Tournament

During the tournament, priority bits are propagated two hops away. This is done by performing the transmission of each bit in two stages. In the first stage - *Transmission stage*, each node transmits its own priority bit. In the second stage - *Retransmission stage*, nodes retransmit the priority bit detected at the first stage. If a node transmitted or detected a dominant bit in one of the two priority bit transmission stages, then it knows that the current priority bit was dominant.

Algorithm 4.1 details this approach. Nodes execute this algorithm (Algorithm 4.1) for each priority bit. Function `prioBitTxStage()` accepts the value of the priority bit

Algorithm 4.1 2-stage priority bit transmission.

Require: Nodes have established a common time reference prior to the transmission of priority bits.

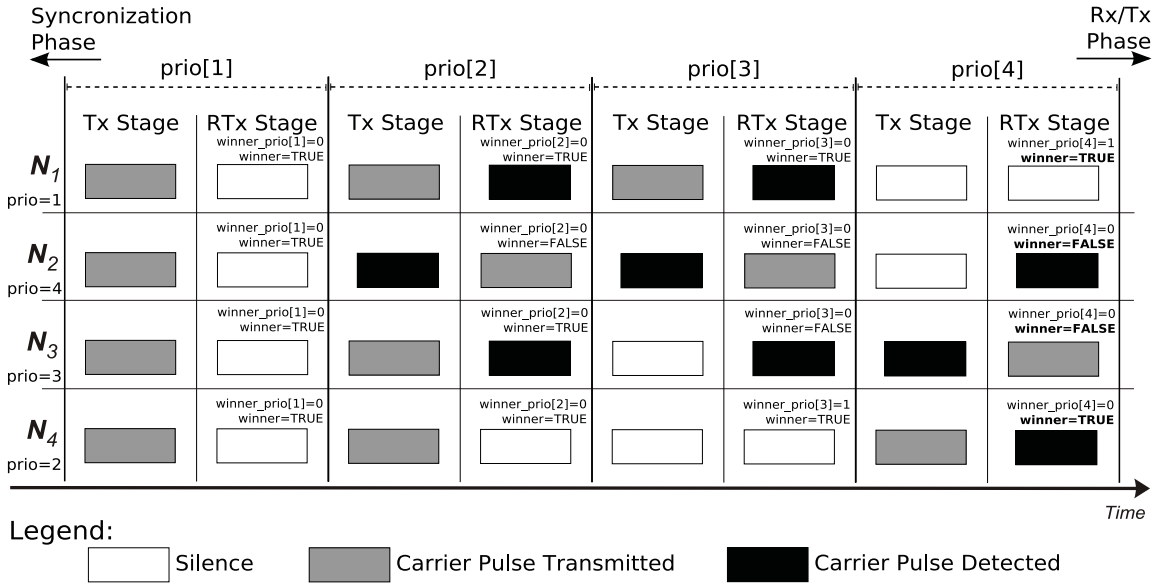
```
1: begin
   {Priority bit transmission stage}
2:   prioBitTx ← call prioBitTxStage(prio[i])
   {Priority bit retransmission stage}
3:   prioBitRTx ← call prioBitTxStage(prioBitTx)
   {Decision}
4:   winner_prio[i] ← prio[i];
5:   if (prioBitTx = 0 OR prioBitRTx = 0) AND (prio[i] = 1) then
6:     winner_prio[i] ← 1;
7:     winner ← FALSE
8:   endif
9:   end
```

to be transmitted and returns the priority bit value detected (if the node had a recessive bit) or transmitted (the node itself transmitted a dominant bit). This function only transmits a dominant bit if the variable `winner` is equal to `TRUE`, because nodes only send their priority bits while they are potential winners.

During the transmission stage (Algorithm 4.1, line 2), the value of the current priority bit is used. In the retransmission stage, the value returned in the previous call to by `prioBitTxStage()` (Algorithm 4.1, line 3) is used. At the end of the two stages, nodes know the winning priority observed and if they lost the tournament for that priority bit (Algorithm 4.1, lines 4 to 9).

Figure 4.4 illustrates a tournament between four nodes with $npriobits = 4$. Nodes N_1, N_2, N_3 and N_4 are accessing the medium with priorities 1, 4, 3 and 2, respectively. Nodes are assumed to have achieved synchronization before starting the transmission of priority bits, and the synchronization error is ignored in this example. Observe that in priority bit 2 (`prio[2]`), N_2 detects a dominant bit during the transmission stage, which causes it to send a carrier pulse in the retransmission stage and to lose the tournament (N_2 sets its winner variable, which indicates if the node is still a possible winner of the tournament to `FALSE`). In bit 3 (`prio[3]`), N_2 detects again a dominant bit during the transmission stage. When N_2 performs the retransmission of this bit,

(a) Network Topology for Example in Figure 4.4b.



(b) Tournament between nodes N_1, N_2, N_3 and N_4.

Figure 4.4: Tournament Example, with Priority Bits Retransmission.

N_3 will detect it and will lose the tournament.

Note that at the end of the tournament, two nodes, N_1 and N_4, have `winner=TRUE` and thus will both transmit a message. Nodes N_1 and N_4 behave correctly, as they do not share any common receiver. This illustrates an important characteristic of our protocol: it allows multiple winners, and thus parallel transmissions are allowed.

4.3.2 Details of the Protocol

In this section, the full protocol for MBD wireless networks is presented, discussing the protocol's properties and correctness.

The proposed protocol is also (like the version for SBD proposed in Chapter 3) composed of three main phases: *synchronization*, *tournament* and *receive/transmit*. The full protocol is formally presented in two figures using the same timed-automata style notation as used in Chapter 3. States are numbered from 0 to 17. State 0 is the initial state. Figure 4.5 presents the details of the synchronization phase, while

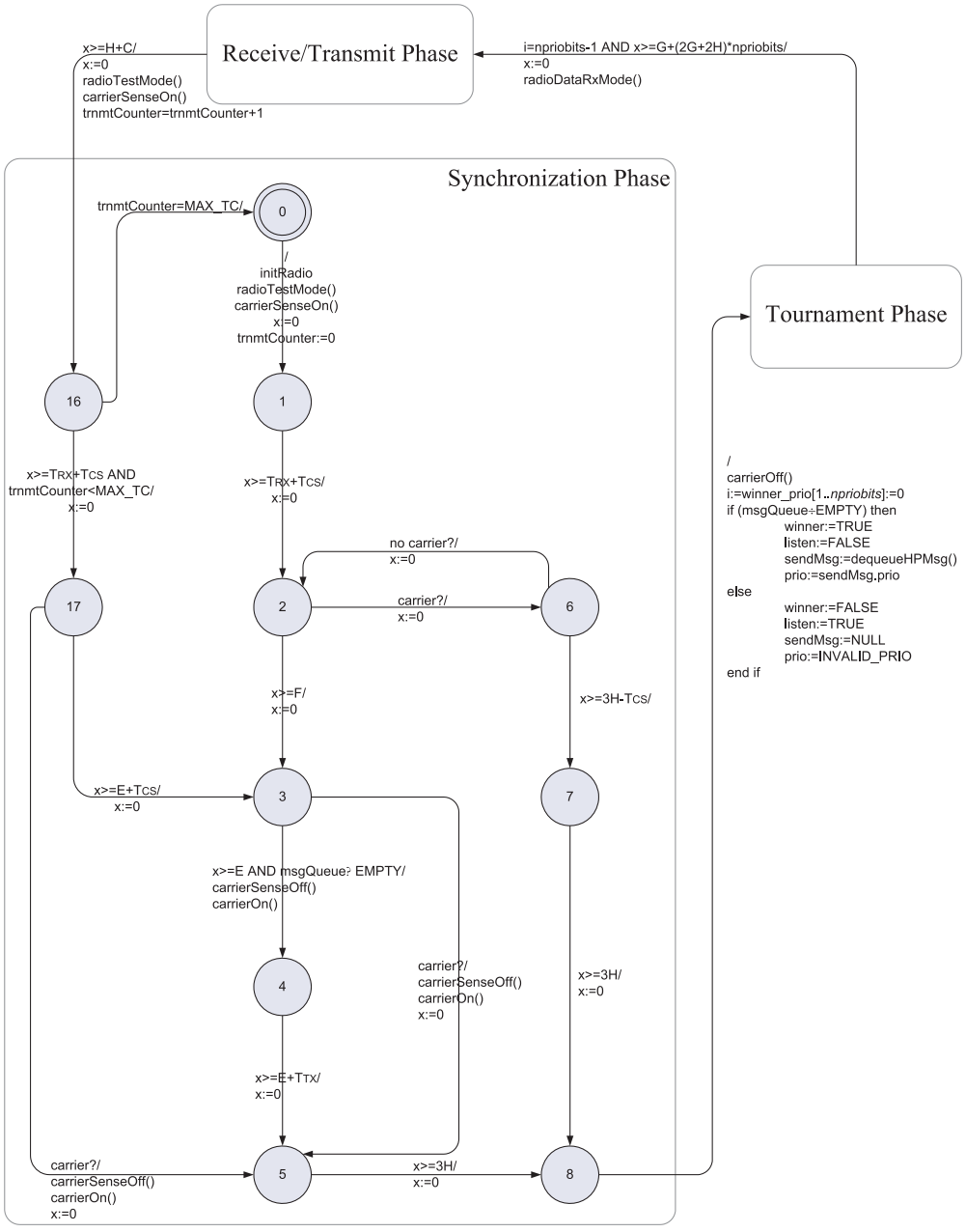


Figure 4.5: WiDom-MBD Time Automaton - Synchronization Phase.

Figure 4.6 presents the details of the tournament and receive/transmit phases.

The desciption follows a trace through a simple sequence of state transitions that nodes go through in order to synchronize with their 2-neighbors after they boot. After initializing the radio, nodes move into State 1. Transition $1 \rightarrow 2$ ensures that the

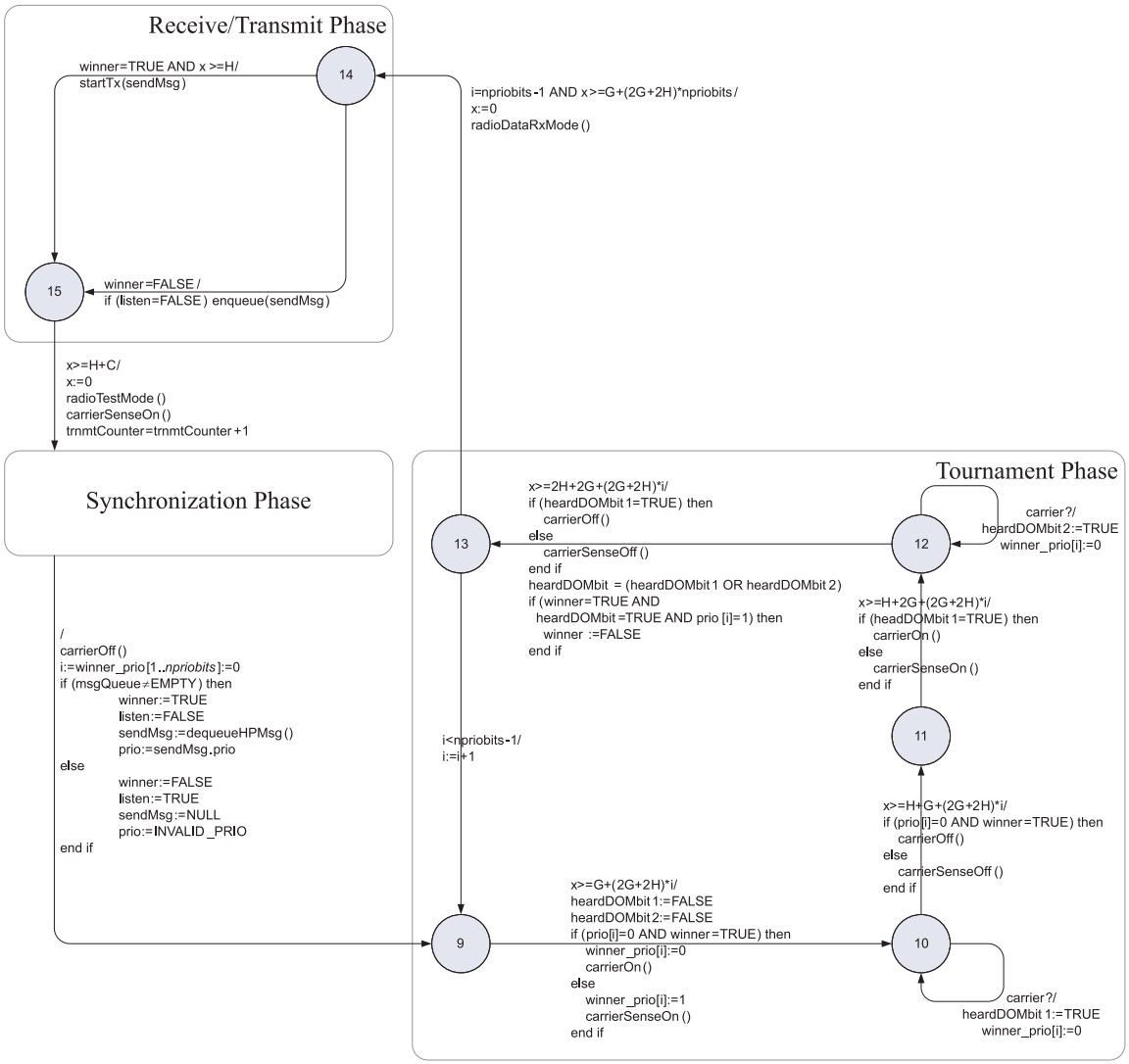
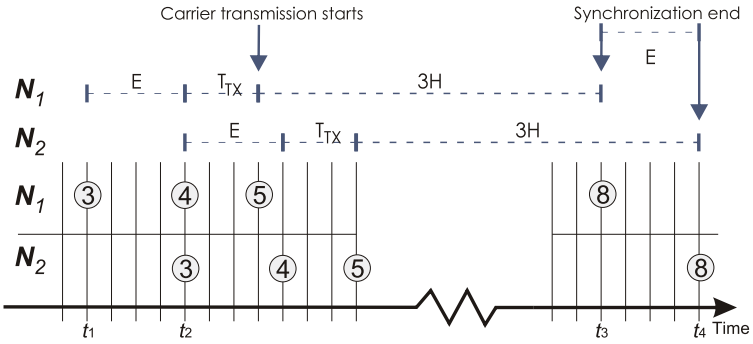


Figure 4.6: WiDom-MBD Time Automaton - Tournament and Rx/Tx Phases.

radio changes to receive mode and monitors the medium for an amount of time long enough to detect if the medium is idle. In State 2, nodes wait for a long period of silence (denoted by F), such that no node disrupts a tournament taking place by other nodes. Next, nodes with pending message requests will perform transition $3 \rightarrow 4$ after waiting for E time units, so that other nodes have time to reach State 3. Nodes that make the transition $3 \rightarrow 4$ start sending a carrier pulse in order to synchronize. Other nodes may take one of the two following sequence of state transitions: (i) a node is in State 3 and has pending messages and it did not hear a carrier for E time units so

Figure 4.7: Topology for Scenarios in Figures 4.8, 4.9 and 4.10.

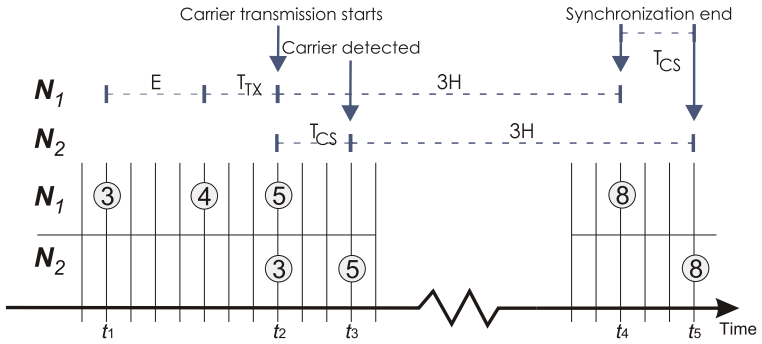


(a) N_1 and N_2 Initiate a Tournament.

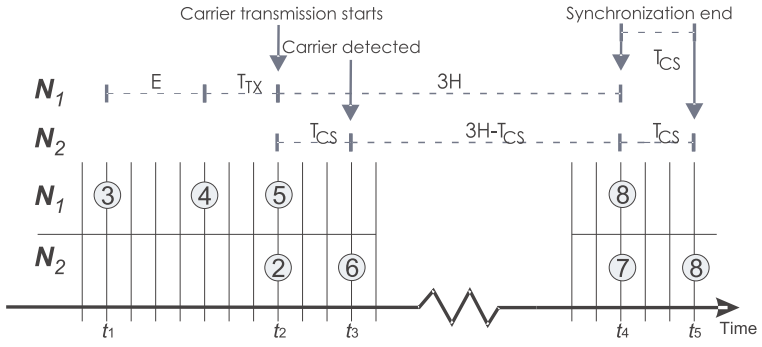
Figure 4.8: Synchronization Scenario 1.

it makes the transition $3 \rightarrow 4$; or (ii) a node in State 3 (either because it is waiting to make transition $3 \rightarrow 4$, or it does not have any pending messages) can detect the carrier pulse being sent by other nodes and performs transition $3 \rightarrow 5$. Nodes making transition $3 \rightarrow 5$ start transmitting the synchronization carrier pulse and immediately reset their timers. Meanwhile, nodes making transition $3 \rightarrow 4$ wait T_{TX} time units to reset their timers because only at that time the carrier pulse is actually being transmitted. Nodes then stay in State 5 sending the synchronization carrier pulse and make transition $5 \rightarrow 8$ after $3 \times H$ time units (the length of this pulse was selected such that it is a multiple of the duration of a bit in the tournament and is long enough to guarantee reliability in its pulse, and synchronization ends with nodes resetting their timers).

Nodes can actually take a number of different sequences of state transitions to synchronize. In the remainder of this section, these transitions will be discusses, as well as the resulting synchronization error.



(a) N_1 Initiates a Tournament; N_2 Follows Departing From State 3.



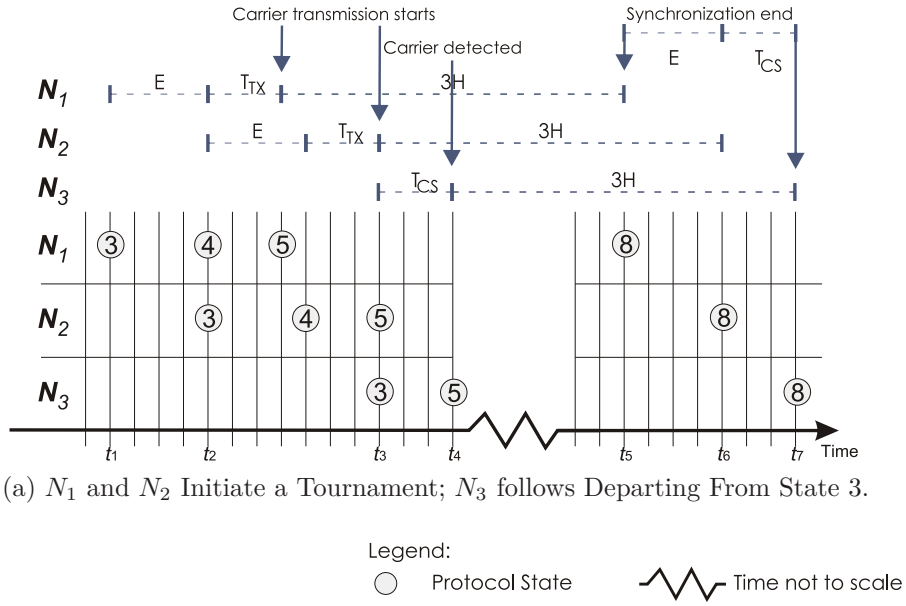
(b) N_1 Initiates a Tournament; N_2 Follows Departing from State 2.

Figure 4.9: Synchronization Scenarios 2 and 3.

Synchronization Error

As previously mentioned, the synchronization error influences the duration of each priority bit (H) in the tournament and the minimum time interval (G) between them. To analyse and study the occurrence of synchronization error, the possible scenarios to achieve synchronization among 2-neighbor nodes that may result in the worst-case synchronization error are considered. Figures 4.8-4.10 illustrate these scenarios for the same network topology as depicted in Figure 4.7.

Figure 4.8 illustrates the first scenario where two neighbor nodes N_1 and N_2, have both pending messages to transmit. Node N_1 enters State 3 at time t_1 where it remains for E time units (to ensure that other nodes have time to reach State 3). In a worst-



(a) N_1 and N_2 Initiate a Tournament; N_3 follows Departing From State 3.

Figure 4.10: Synchronization Scenario 4.

case scenario, at time t_2 node N_2 will enter State 3 exactly at the same time node N_1 leaves State 3. If we choose E such that $T_{CS} \leq E \leq T_{TX} + T_{CS}$, then N_2 will never detect the carrier being sent out by node N_1, and thus N_2 will proceed to State 4. Both nodes will do exactly the same transitions, but with E time units of difference between them. When the nodes finally finish synchronization (reaching State 8) at times t_3 (N_1) and t_4 (N_2), the synchronization error will be at most E.

Another scenario in Figure 4.9 illustrates the state evolution that two nodes (N_1 and N_2) experience when only node N_1 has pending messages. Both figures (4.8 and 4.9) show how nodes can enter State 8 with a maximum difference of T_{CS} time units.

The synchronization scenarios depicted in Figures 4.8 and 4.9 concern the synchronization between two directly connected nodes. Figure 4.10 considers a third node (N_3) that is a 2-neighbor of N_1 and N_2. Nodes N_1 and N_2 perform the same sequence of state transitions as in Figure 4.8. Node N_3 detects the retransmission of the carrier pulse started by node N_2 at time t_4. Consequently, N_3 reaches State 8 T_{CS} time units after time t_6, when node N_2 reached State 8 and $E + T_{CS}$ after node N_1 that reached State 8 at time t_5. The sequence of state transitions made by node N_3 in this scenario

is similar to the one made by N_2 in Figure 4.9a, and likewise node N_3 could be in State 17 $E + T_{CS}$ time units before time t_4, and take transition $17 \rightarrow 5$. Observe that N_3 can follow a sequence of state transitions similar to node N_2 in Figure 4.9b, and thus would reach State 8 T_{CS} time units after N_2 and $2 \times T_{CS}$ after node N_1.

Following from the previous discussion, the maximum synchronization error (δ) among 2-neighbor nodes is given by:

$$\delta = \max\{E + T_{CS}, 2 \times T_{CS}\} \tag{4.1}$$

Error Mitigation

According to the time automaton in Figures 4.5 and 4.6, the normal behavior of a node after sending/receiving a data message (in State 16) is to proceed to synchronization without waiting to observe a long period of silence. If nodes only waited for a long period of silence after booting up, then a single synchronization failure could compromise the network for an arbitrarily long period of time. To avoid this, nodes periodically perform transition $16 \rightarrow 0$, forcing them to wait for a long period of silence. This transition is made every time a node performs `MAX_TC` tournaments. The value `MAX_TC` can be adjusted to the radio environment and transceivers. In our experiments, we have, `MAX_TC` = 100.

4.3.3 Rationale of the Design and Correctness

It is necessary to select timeout parameters to ensure that synchronization before the tournament works properly. In this section, we discuss the correctness of the protocol behaviour and demonstrate how assigning values to the constants C, E, F, G, H, T_{CS}, T_{TX} and T_{RX} affect that correctness. The protocol must satisfy the following three relevant properties (P4.1-P4.3):

Property 4.1 Mutual Exclusion. *There is no pair of nodes (N_i, N_j) such that: (i) N_i is a 2-neighbor of N_j and (ii) N_i and N_j determine to be winners of the tournament*

(i.e. both nodes are in State 15 and the variable **winner** *in N_i and N_j is simultaneously* TRUE*).*

Property 4.2 Progress. *Consider a node N_i that requests to transmit. For every 2-neighbor node N_j of N_i such that $prio_{N_j} < prio_{N_i}$, it holds that at most Q_{HP} time after the request to transmit, node N_i ends the tournament and advances to the receive/transmit phase as winner (i.e. node N_i is in State 15 and the variable* **winner** *is equal to* TRUE *Q_{HP} time after the request to transmit). Q_{HP} will be derived next.*

Q_{HP} can be derived from inspection of the time automaton in Figures 4.5 and 4.6. It is the maximum time that the highest priority message may wait until its transmission starts. This includes waiting for an ongoing tournament to finish (the time to evolve from State 3 to State 15: $3H+(npriobits-1)\times(2G+2H)+G+H$), the maximum time for transmitting a message (C), and the time to evolve from State 0 (thus assuming that transition $16 \rightarrow 0$ is made) to State 15, $C + T_{TX} + T_{CS} + F + 3H + (npriobits - 1) \times (2G + 2H) + G + H$. Thus, Q_{HP} is given by:

$$\begin{aligned} Q_{HP} = {} & C + T_{TX} + T_{CS} + F \\ & + 2 \times (3H + (npriobits - 1) \times (2G + 2H) + G + H) \end{aligned} \tag{4.2}$$

Using the same reasoning as above, $Q_{trnmt-MBD}$ can defined as the time to perform a tournament considering the overhead of the protocol when nodes are not synchronized:

$$\begin{aligned} Q_{trnmt-MBD} = {} & T_{TX} + T_{CS} + F \\ & + 3H + (npriobits - 1) \times (2G + 2H) + G + H \end{aligned} \tag{4.3}$$

This will be the amount of time to minimize when trying to instantiate the protocol parameters.

Property 4.3 Prioritization. *If a node N_i requests to transmit and node N_i is in State 15 with its variable* **winner** *equal to* FALSE*, then there is a node N_j such that (i)*

N_j is a 2-neighbor of N_i, (ii) N_j has requested to transmit and (iii) $prio_{N_j} > prio_{N_i}$.

These properties hold if the following constraints (4.1-4.7) are satisfied.

Constraint 4.1 *When a node transmits a dominant bit in iteration i in the tournament, this dominant bit is received by all other nodes and all other nodes perceive this dominant bit to be in iteration i.*

Consider an iteration of the tournament. Sufficient time overlap between the transmission of a carrier and the time interval where a node with a recessive bit listens must exist. Due to clock drift and inaccuracy of synchronization, the last iteration of the tournament (the worst scenario) is considered in the following constraint:

$$\begin{aligned}(3H + H + G + (2H + 2G) \times (npriobits - 1)) \times (1 - \varepsilon) \\ -(3H + G + (2H + 2G) \times (npriobits - 1)) \times (1 + \varepsilon) \\ -2CLK - L - 2\alpha - \delta > T_{CS} + 2 \times T_{RX}\end{aligned} \tag{4.4}$$

Inequality 4.4 implies that even in the presence of worst-case clock inaccuracies, all nodes will hear a dominant bit for at least the time necessary to detect a carrier (T_{CS}).
□

Constraint 4.2 *If a node N_i has perceived a time of silence long enough (F time units) to make the transition $2 \rightarrow 3$ but other nodes perceive the duration of silence to be less than F time units so far (due to different time-of-flights and clock-imperfections), then node N_i must wait until all nodes have perceived this long time of silence.*

If inequality (4.5) is true

$$2CLK + L + 2 \times \alpha + (F + T_{RX} + T_{CS}) \times 2\varepsilon + T_{CS} < E \tag{4.5}$$

then the protocol must stay in State 2 for E time units, and this ensures Constraint 4.2 is true.
□

Constraint 4.3 *With similar reasoning as for Constraint 4.2, a node which has won the tournament must wait H time units before transmission (this waiting occurs in $14 \rightarrow 15$) to guarantee that all losing nodes have reached State 15.*

If inequality (4.6) is true

$$[3H + H + G + (2H + 2G) \times (npriobits - 1)] \times 2\varepsilon + 2CLK + L + 2\alpha + \delta < H \quad (4.6)$$

then Constraint 4.3 is true.

□

Constraint 4.4 *During the tournament, the maximum time interval of idle time should be less than F, the initial idle period.*

If inequality (4.7) is true

$$\begin{aligned} \left[\begin{array}{c} 3H + H + G + (2H + 2G) \times (npriobits - 1) \\ +G + H + C + T_{RX} + T_{CS} + E + T_{CS} \end{array} \right] \times (1 + \varepsilon) \\ -3H \times (1 - \varepsilon) + 2CLK + L + 2\alpha + T_{CS} < F \end{aligned} \quad (4.7)$$

then the amount of silence during a tournament is less than F and hence Constraint 4.4 is true.

□

Constraint 4.5 *The time interval between two successive dominant bits must be long enough to assure that no node interprets the first dominant bit to be transmitted in the time interval for the second dominant bit.*

The worst case occurs when these two consecutive bits are the last ones in the tournament. Therefore, if inequality (4.8) is true

$$\begin{aligned} (3H + H + G + (2H + 2G) \times (npriobits - 1)) \times (1 - \varepsilon) \\ -(3H + G + (2H + 2G) \times (npriobits - 1)) \times (1 + \varepsilon) \\ -2CLK - L - 2\alpha - \delta > 0 \end{aligned} \quad (4.8)$$

then Constraint 4.5 is true.

□

Constraint 4.6 *Transition* $6 \rightarrow 7$ *cannot occur when a node is transmitting a message (a message transmission is detected as a carrier, if nodes are performing carrier detection).*

If inequality (4.9) is true

$$3H \geq C + T_{RX} + T_{CS} \tag{4.9}$$

then Constraint 4.6 is true.

□

Constraint 4.7 *Transition* $15 \rightarrow 16$ *takes, at least, the time to transmit/receive the longest message in the network.*

If inequality (4.10) is true

$$\forall i \in \{1..n\} : C \geq max\{C_i\} \tag{4.10}$$

then Constraint 4.7 is true.

□

The values of E, F, G and H must be selected such as they satisfy Constraints (4.1)-(4.7). The selection of T_{CS}, T_{RX} and T_{TX} is imposed by the implementation platform chosen. Section 3.4.4 instantiates these parameters for a concrete platform.

4.4 Implementation and Evaluation

A prototype implementation of WiDom for MBD as described previously in this chapter was also developed. The protocol was implemented in Nano-RK [110], instead of TinyOS as in Chapter 3. Nano-RK is a reservation-based real-time operating system (RTOS), supporting fixed-priority preemptive multitasking and bandwidth reservations for both CPU and network [110]. Nano-RK runs on both the MicaZ [107] and

Table 4.1: WiDom-MBD Parameters (FireFly Sensor Network Platform).

Parameter	Value	Notes
$npriobits$	10	Developer/Platform-defined parameter
α	1 μs	Defined according to expected node geographical distribution
L	5 μs	Platform-defined parameter
CLK	1 μs	Platform-defined parameter
ε	10^{-5}	Platform-defined parameter
T_{TX}	192 μs	Platform-defined parameter
T_{RX}	256 μs	Platform-defined parameter
T_{CS}	256 μs	Platform-defined parameter
C	2 176 μs	Platform-defined parameter
`MAX_TC`	100	Developer/Platform-defined parameter
F	48 396 μs	Computed from protocol constraints
E	338 μs	Computed from protocol constraints
H	1 443 μs	Computed from protocol constraints
G	675 μs	Computed from protocol constraints
$Q_{trnmt-MBD}$	93 415 μs	Computed from protocol constraints

FireFly [80] sensor network platforms. The development of the protocol in Nano-RK allowed a better control of the timming of the protocol.

A simulation model was also developed. The simulation enables the study of the protocol's overall behavior in medium sized networks (we have run our simulation experiments with 30 nodes). The simulation was also used to study the protocol behavior under controlled adverse conditions.

4.4.1 Instantiating the Protocol Parameters.

The protocol parameters must be selected according to the constraints (4.1)-(4.7) as defined in Section 4.3.3, and the platform-dependent parameters, are given in Table 4.1 (these parameters are for the FireFly sensor network platform). The time to transmit a message in the CC2420 [108] is computed assuming that the data length of messages is 64 bytes, as reasoned in Section 3.4.

The transceiver switching times were derived from the datasheet. It takes 128μs for the transceiver to switch to receive mode, added to the time until the first CCA

operation (128μs). The transceiver takes 192μs to switch to transmit mode. Note that, different from the implementation in Chapter 3, the parameters are measured at runtime in μs, because the implementation used a timer with such granularity ($CLK = 1\mu$s). The value of T_{CS} was determined experimentally. Note that while the same radio transceiver as the implementation is Chapter 3 was used, the FireFly sensor network platforms have better antennas than the MicaZ and this allowed to reduce T_{CS}.

Parameter L, the time for the execution of the protocol, was defined to allow for about 40 instructions to be executed on the microcontroller for each state transition of the protocol. Parameter `MAX_TC` was discussed in Section 4.3.2.

Note that, the value (113 789μs) of Q_{HP} in [8] was obtained for $npriobits = 5$ and $T_{CS} = 486\mu$s. For the values presented in Table 4.1, $Q_{HP} = 137\ 986\mu$s.

A choice of parameters F, E, H, G that satisfies constraints (4.1-4.7) was obtained simply by formulating the following optimization problem:

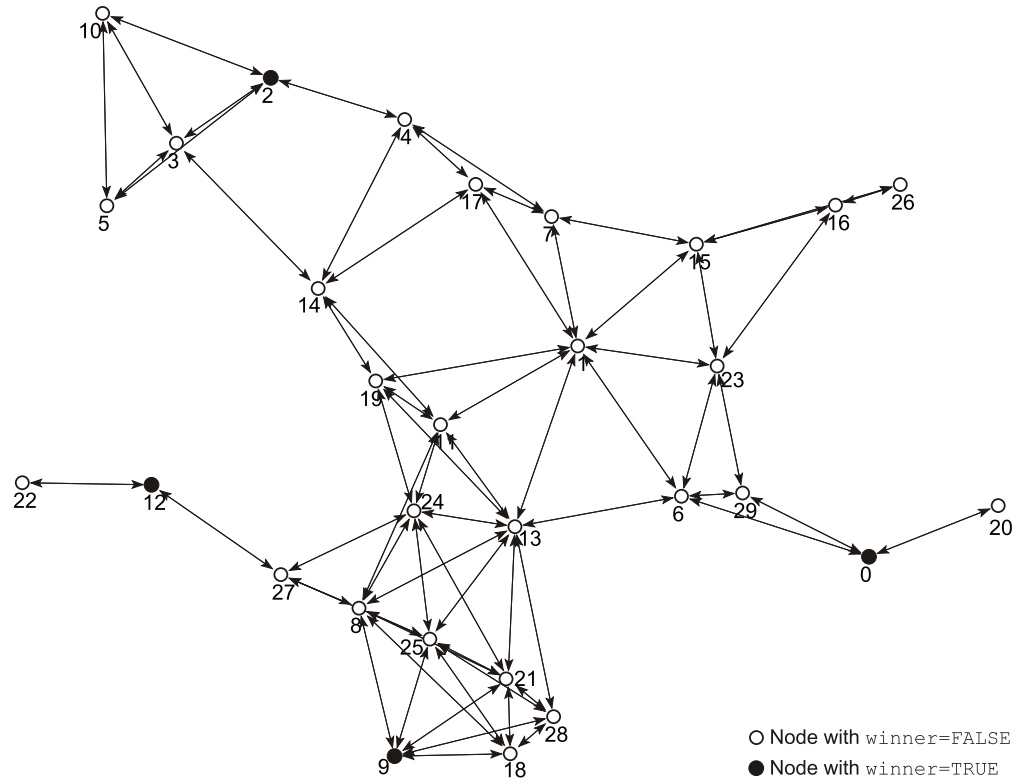
minimize $Q_{trnmt-MBD}$**, *subject to Constraints* 4.1-4.7.**

$Q_{trnmt-MBD}$ is the time to perform a tournament considering the overhead of the protocol when nodes are not synchronized, therefore this is the amount of time to be minimized for all message streams.

A summary of the parameters obtained for this implementation using the FireFly sensor network platform are presented in Table 4.1.

4.4.2 Simulation Results

The protocol was also implemented in a discrete event simulator. This simulation model enables studying the protocol behavior under different error conditions (in this case, different probabilities of detecting a synchronization carrier or a priority bit). The simulation implements the time automaton as shown in Figures 4.5 and 4.6. Using $npriobits = 5$, the protocol was tested through 100 independent simulation runs for each scenario with varying probabilities of missing the detection of a carrier pulse.



(a) Start of tournament. (b) End of bit 1 transmission. (c) End of bit 2 transmission.

(d) End of bit 3 transmission. (e) End of bit 4 transmission. (f) End of bit 5 transmission.

Figure 4.11: Example Topology Graph Illustrating Parallel Transmissions and Tournament Evolution in WiDom-MBD.

Each node was setup with one message stream having a unique priority between [0,29] and an exponentially distributed message inter-arrival time, with an expected value

ranging between 0.01 and 1 second. In this simulation model, tyhe initial study is focused on the behavior of the MAC protocol, so messages are just broadcast once from its original source node.

For the simulation, the values for the timeouts C, E, F, G, H, T_{CS}, T_{TX} and T_{RX} were instantiated for a platform with parameters found in literature [10, 11]. Assuming a radio with a maximum range of 30 m, we have $\alpha = 0.1\mu$s. Typical microcontrollers have $CLK = 1\mu$s and $\delta = 10^{-5}$. Assuming that the protocol is implemented on dedicated hardware, $L = 1\mu$s. The value of $T_{CS} = 5\mu$s was chosen because busy tone detection of narrow-band signals can be achieved in this amount of time [11], and our application of carrier sensing is similar to busy tone detection. We assume $T_{TX} = T_{RX} = 1\mu$s; such transceivers have been implemented [10]. Let $npriobits = 5$. One choice that satisfies the constraints in Section 4.3.3 for these parameters is: $E = 10\mu$s; $F = 553\mu$s; $G = 20\mu$s; $H = 30\mu$s. Messages are assumed to have at most 54 bytes [10]; at a data rate of 36 Mb/s, $C = 12\mu$s.

Each simulation run used a random topology with 30 nodes, where each node has on average 3 direct neighbor nodes. An example of a topology generated by the simulator is illustrated in Figure 4.11. The numbers aside each node (circle) represent the priority given to the message stream of that node. The topology was constructed by randomly placing nodes within a bounded area and maintaining a minimum distance between nodes. Connectivity depends on if the power level at the receiver is above a defined threshold. The power level at the receiver is calculated as a function of the distance between nodes, using a log-normal shadowing model [12] with parameters $P_t = 0$dBm, $G_t = G_r = 1$dBi, $d_0 = 1$m, $\lambda = 0.125$m (assuming a 2.4 GHz operating frequency), $n = 2.5$ and $\sigma = 5$.

In all simulation runs, nodes perform more than 50 000 tournaments. After each tournament, it was detected whether the correctness properties collision-free, progress and prioritization were satisfied for all nodes in the network. Tournaments where any node in the network failed to satisfy one of the properties are named erroneous tournaments. These erroneous tournaments were caused by either failure to detect a

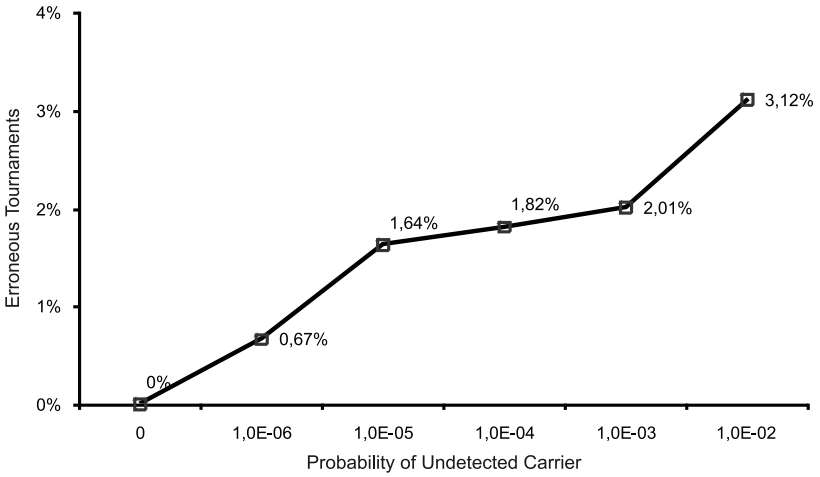


Figure 4.12: Probability of an Erroneous Tournament.

synchronization carrier, or a priority bit. The probabilities of observing an erroneous tournament are plotted in Figure 4.12. The fact that no errors were found with a perfect detection of carriers presents evidence that the protocol correctness properties are satisfied. Observe also that the error ratio is still under a low value when nodes fail to detect carriers.

4.4.3 Example and Discussion of Parallel Transmissions

As shown in Figure 4.4, our protocol allows parallel transmissions. The simulated example shown in Figure 4.11 is a larger example to exemplify this. Figure 4.11 shows a network topology with 30 nodes. The numbers aside each node (circle) represents the priority given to the message stream of that node. The figure also depicts the result of a tournament where all nodes requested to transmit. With this example, we observe the 4 parallel transmissions (the nodes winning a tournament are marked with a solid black circle) allowed by the protocol.

Figure 4.11 illustrates the progression of the tournament for this scenario. Figure 4.11a shows that at the beginning of the tournament all nodes are potential winners, and Figures 4.11b through 4.11e illustrate the nodes that were potential winners at the end of transmission of priority bits 1 to 5 during the tournament. Observe that

node with priority 26 does not win the tournament, but at the same time, neither do any of its 2-neighbors. This happens because node 15 is a 2-neighbor to node 26 and it causes node 26 to lose in the first priority bit of the tournament. Later in the tournament, node 15 lost as well (observe the sequence of Figures 4.11a to 4.11c). This phenomenon is known from previous research and has been dubbed multihop competing [102].

4.5 Conclusions

This chapter introduced a MAC protocol that is prioritized and collision-free in networks with MBD in the presence of hidden nodes. It achieves prioritization without base stations and without relying on out-of-band signals.

A protocol that provides an upper bound on the queuing times of messages is naturally useful for supporting scheduling of real-time traffic. While this is not sufficient to provide hard real-time guarantees in practice, this protocol can be a useful building block for wireless real-time systems and offers a solid foundation for schedulability analysis techniques for wireless networks (such as [64]).

The proposed protocol was implemented and tested to show that the correctness properties stated are not violated.

It is noted that the overhead introduced by the protocol is, to a large extent, due to the transmission/reception switching time and the time necessary to perform carrier sensing. This is a technological limitation that can be overcome with better hardware. This aspect will be addressed in Chapter 6.

Chapter 5

Computing Aggregate Quantities by Exploiting WiDom

Contents

5.1 Introduction

One of the motivations for proposing the use of a dominance-based MAC protocol to efficiently obtain aggregate values in large-scale, dense wireless sensor networks is the property of those MAC protocols that collisions are non-destructive.

By choosing the priority field for accessing the medium dynamically in each node, it is possible to exploit a prioritized medium access control (MAC) protocol to obtain certain aggregated quantities in a SBD in a scalable and efficient way. Here, "scalable" means that consumption of resources and the time complexity increases slowly or not at all as the number of nodes increases.

In this chapter, several distributed algorithms to obtain aggregate quantities that exploit these features are presented. The minimum value (MIN) can be obtained with a time-complexity of $O(npriobits)$, where $npriobits$ is the number of bits used to represent the data (for example, sensor data). Note that, in this case, the message complexity (and thus, the time-complexity) is independent of the number of sensor nodes. The same technique can be applied to obtain the maximum value (MAX). It is also shown how to obtain more complex aggregated quantities such as MEDIAN, COUNT and Interpolation.

The distributed algorithms developed for SBDs can be adapted to be used in MBD systems. In that context, it is proposed an algorithm for computing the MIN or MAX of sensor readings in a multihop network. This algorithm has a time-complexity that does not depend on the number of sensor nodes; its complexity only depends on the network diameter and the range of the value domain of sensor readings.

5.2 Assumptions, Notation and MAC Interface

Throughout this chapter the assumptions and notation in use are those already described earlier in Section 3.2. Additional notation and assumptions regarding priorities, sensor readings and node location will also be used as follows:

Priorities. Let MAXP denote the maximum value of the priorities; that is MAXP $= 2^{npriobits} - 1$.

Sensor Readings. A node N_i takes a sensor reading s_i as a non-negative integer in the range [0, MAXS], where MAXS is defined by the length of the ADC on the platform. Considering that $NADCBITS$ is the number of bits of the ADC on the platform, MAXS $= 2^{NADCBITS} - 1$.

Node Location. The interpolation technique in Section 5.3.5 requires that nodes know their location. The location of a node N_i is defined by its two coordinates (x_i,y_i). The function $f(x,y)$ denotes the interpolation of the sensor data at the coordinates x_i,y_i.

MAC Protocol Programming Interface

The system calls and functions that can be considered as part of the MAC protocol programming interface are overviewed next.

A program on a node can access the communication system via the following interface. The `send` system call takes two parameters, one describing the priority of the packet and the other describing the data to be transmitted. If a node calling `send` wins the contention, then it transmits its packet and the program making the call unblocks. If a node calling `send` loses the contention, then it waits until the contention resolution phase has finished and the winner has transmitted its packet (assuming that the winner did not send an empty packet). Then, the node contends for the channel again. The system call `send` blocks until it has won the contention and transmitted a packet. The function `send_empty` only takes a priority as parameter, causing the node to perform the contention without sending any data after the contention.

In addition, when the contention is over (regardless of whether the node wins or loses), the function `send_empty` gives the control back to the application and returns the priority of the winner.

The system call `send_and_rcv` takes two parameters, a priority and data to be

transmitted. The contention is performed with the given priority and then the data is transmitted if the node wins. Regardless of whether the node wins or loses, the system call returns the priority and data transmitted by the winner and then unblocks the application.

In implementations using TinyOS [109], the semantics of these calls was changed to be according to the split phase semantics typically used in this operating system. For example, a `send` call will return immediately, and, when the contention resolution phase has finished, a call back function will return the result.

5.3 Algorithms for the SBD Case

First, the focus is given on obtaining aggregate quantities on a SBD. By exploiting WiDom, it is straightforward to show that MIN can be obtained by performing one tournament where all nodes participate using their proposed value as priority. MAX can be obtained similarly.

Based on the technique to obtain MIN, a more complex aggregated quantity can also be obtained: MEDIAN. This computation hinges on the ability to compute the number of nodes (COUNT). Such a technique is presented, but it only gives an estimation and hence, the median function is only estimated.

It is often required to know how physical quantities (such as temperature) vary over an area or region. Clearly the physical location of each node must be known beforehand. For such systems, an algorithm that produces an interpolation of the sensor data as a function of space coordinates is proposed. This interpolation is a compact representation of sensor data at a moment and it can be obtained efficiently.

5.3.1 Computing MIN

Consider a simple application where a node (node N_1) needs to know the MIN of the temperature readings among its neighbors. Suppose that the temperature values are

coded as n-bit integers. Starting with the most significant bit first, let each node send the temperature reading bit-by-bit. Consider also that, for each transmitted bit, nodes read the resulting value in the channel (something straightforward in a wired medium) and the channel implements a logical AND of the transmitted bits. Furthermore, if a node reads '0' and is transmitting a '1', it stops transmitting. Then at the end of the transmission of n bits, the "observed" value in the channel will correspond to the MIN. It is as if all m temperature readings were transmitted in parallel.

The above behavior of the channel can be achieved by WiDom. And thus, we can exploit WiDom to compute MIN with a time-complexity that is equivalent to the time of transmitting a single message.

The algorithm to compute MIN is formally presented in Algorithm 5.1. Function `send_empty` causes the node to perform the contention for the medium and returns the priority of the winner (See Section 5.2).

Algorithm 5.1 Computing MIN.

Require: All nodes start Algorithm 5.1 simultaneously.

```
1: function comp_min( ) return the minimum value
2:     v_i ← value proposed by the node (for example, its sensor value)
3:     return send_empty( priority = v_i )
4: end function
```

5.3.2 Computing MAX

MAX can be computed in an analogous way to MIN. Each node uses the bitwise negation of its sensor reading as a priority and competes for the channel. The MAX is then obtained as the bitwise negation of the winning priority, as expressed in Algorithm 5.2.

Algorithm 5.2 Computing MAX.

Require: All nodes start Algorithm 5.2 simultaneously.

```
1: function comp_max( ) return the maximum value
2:     v_i ← value proposed by the node (for example, its sensor value)
3:     return send_empty( priority = bitwise_negation( v_i ) )
4: end function
```

5.3.3 Computing COUNT or the Number of Proposed Values

Using the basic idea of computing MIN introduced in Section 5.3.1, it is possible compute COUNT; that is, the number of nodes. It can also be used to count the number of nodes with a certain attribute. This approach was proposed in [111].

A prioritized MAC protocol allows all nodes to know the priority of the winner but it cannot determine how many nodes had this priority. For this reason, it is not possible to compute the number of nodes exactly; it is only an estimate.

Consider that nodes use random priorities. If the number of nodes is sufficiently large, then the probability approaches 100% for the event that the minimum priority is 0. But if there is only one node, it is highly unlikely that the minimum value among the random priorities is 0. This observation, suggests the possibility of estimating the number of nodes from the minimum of random numbers; this is further described next.

Estimating a Single Value

The pseudo-code of the algorithm for estimating the number of nodes is shown in Algorithms 5.3 and 5.4. All computer nodes start their execution simultaneously and use a global boolean variable `active` as input, indicating if the node should be considered in the COUNT operation. When performing Algorithm 5.3, all nodes have `active` equal to `TRUE` and proceed in following way. First, on line 7, the algorithm generates a random number in the range [0, $MAXP$] ($MAXP$ is the maximum priority value supported; see Section 5.2), then all nodes send their random number. Recall, from Section 5.2, that when nodes call `send_empty`, the priority of the winner is returned and hence all nodes know the minimum random number (line 11). This is performed k times, where k is a configuration parameter that controls the accuracy of the estimate. The line 16 uses a function, shown in Algorithm 5.4 to compute the estimation of the number of nodes based on the minimum numbers obtained on line 11. The `if` statement on line 13 deals with the unlikely event that one of the minimum random numbers is $MAXP$.

Algorithm 5.3 Estimating COUNT (the number of nodes)

Require: All nodes start Algorithm 5.3 simultaneously.
Require: A global boolean variable – *active* – indicating if the node is considered

```
1: function nnodes(x : array[1..k] of integer) return an integer
2: r : array[1..k] of integer
3: q : integer
4: est_nodes : integer
5: for q ← 1 to k
6:     if (active = TRUE) then
7:         r[q] ← random(0, MAXP)
8:     else
9:         r[q] ← MAXP
10:    end if
11:    x[q] ← send_empty(r[q])
12: end for
13: if (∃q : x[q] = MAXV) then
14:    est_nodes ← 1
15: else
16:    est_nodes ← ML_estimation(x[1], x[2], ..., x[k])
17: end if
18: return est_nodes // the estimation of COUNT
```

Algorithm 5.4 Function `ML_estimation`

Require: The division of two integers (as is done in line 6) returns a real number.

```
1: function ML_estimation(x : array[1..k] of integer) return an integer
2:     v : array[1..k] of real
3:     sumv, q : integer
4:     sumv ← 0
5:     for q ← 1 to k
6:         v[q] ← ln( 1 / (1 − x[q]/MAXV) )
7:         sumv ← sumv + v[q]
8:     end for
9:     return ⌈k/sumv⌉
10: end function
```

The design of the function in Algorithm 5.4 can be explained in terms of maximum-likelihood estimation. The details of the estimation can be found in [111].

Estimating an Interval

It is sometimes necessary to estimate an interval or a ranges of values, i.e., to know the probability that the number of nodes is less than or equal to j_2. Or if that number of nodes is greater than or equal to j_1. Since application requirements may vary, a simple generic function is here proposed to compute the *a posteriori probability* that $j_1 \leq m \leq j_2$, where j_1 and j_2 are parameters selected by the application designer. If the probability is not large enough, the application program may decide to decrease j_1 or increase j_2, or perform an estimation with a larger k.

Such algorithm can be developed based on the same principle as the number of nodes estimation presented in the previous section. Such algorithm as well as its rationale is further described in [111].

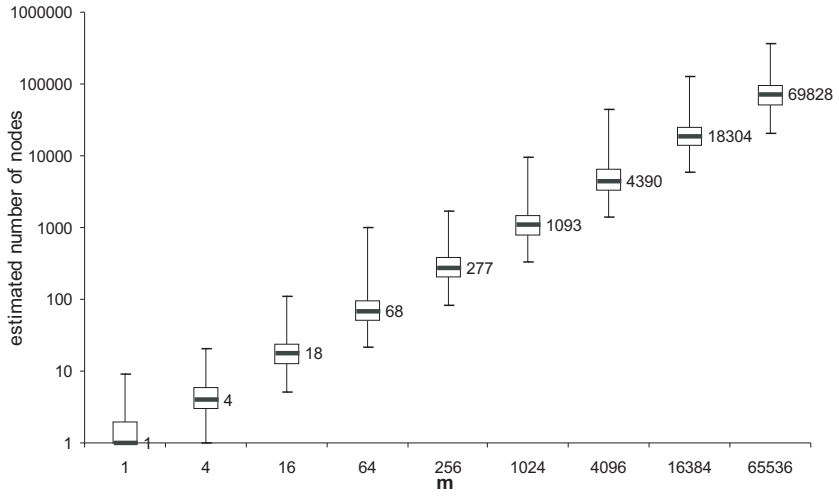
Performance Evaluation

To study how the error of the algorithm depends on m and k, a simulation of the algorithm proposed was developed and executed for $k = 5$ and $k = 20$, and for different numbers of nodes (m = 1, 4, 16, ..., 2^{16}). The box plots in Figure 5.1 are presented in a logarithmic scale and depict the distribution of 1000 estimations for the different numbers of nodes. In these box plots, the box stretches from 25^{th} percentile to the 75^{th} percentile. The value of the median of the 1000 estimations is depicted as a line across the box. Each minimum value is depicted below the box and the maximum value above.

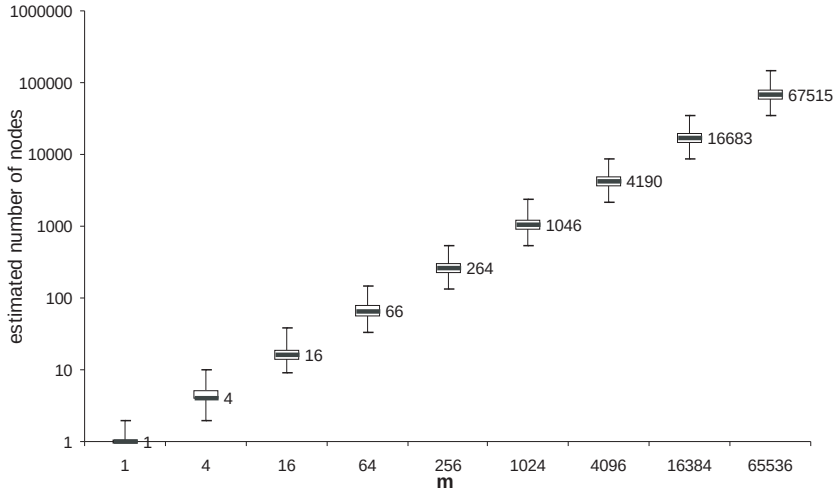
From the box plots in Figure 5.1, it is possible to observe that the quality of the estimation improves significantly by increasing k, and the error is often acceptable for $k = 20$. For large networks, the overhead due to the use of $k = 20$ is also acceptable.

5.3.4 Computing MEDIAN

Consider now the case where the function that we want to compute is the median of v_1,v_2,...,v_m. Let us define $V_{less}(q)$ and $V_{greater}(q)$ as folows:



(a) $k = 5$



(b) $k = 20$

Figure 5.1: Estimation of the Number of Nodes for Different Values of m and k.

$$V_{less}(q) = \{v_j : v_j \leq q\} \tag{5.1}$$

$$V_{greater}(q) = \{v_j : v_j \geq q\} \tag{5.2}$$

With these definitions our goal is to find q such that $||V_{greater}(q)| - |V_{less}(q)||$ is

Algorithm 5.5 Computing MEDIAN

Require: All nodes start Algorithm 5.5 simultaneously.
Require: A global boolean variable – *active* – indicating if the node is considered

```
1: function calcmedian(vi : integer) return an integer
2:     LB ← 0
3:     UB ← MAXP
4:     for j ← 1 to log2(MAXV − MINV) do
5:         mid ← (LB + UB)/2
6:         active ← vi ≤ mid
7:         nVless ← call Algorithm 5.3
8:         active ← vi ≥ mid
9:         nVgreater ← call Algorithm 5.3
10:        if nVless ≤ nVgreater then
11:            LB ← mid
12:        else
13:            UB ← mid
14:        end if
15:    end for
16:    return mid
17: end function
```

minimized. This can be achieved as follows. We know that $0 \leq q \leq MAXP$. For this reason, a lower bound (LB) of q is 0 and an upper bound (UB) of q is $MAXP$. After that, the midpoint (mid) between LB and UB is computed. Then, count the number of nodes with a sensor reading in the interval [LB, mid] and the number of sensor readings in the interval [mid, UB]. If there are more sensor readings in the former than in the latter, then we know that $q \in [LB, mid]$ and hence, UB=mid is set and repeat the argument. Otherwise, $q \in [mid, UB]$ and hence we set LB=mid and repeat the argument. The process ends when LB=UB, providing the median as result. Algorithm 5.5 reflects the pseudo-code of our proposal to the MEDIAN. Note that this result is an estimation, as Algorithm 5.3 is used.

5.3.5 Interpolation of Sensor Data

A wired dominance MAC protocol can also be exploited to track how a physical quantity varies over an area observed by a dense deployment of sensor nodes. The approach

proposed is based on the previously presented approach to compute MIN/MAX, has a time-complexity that does not depend on the number of sensor nodes, and presents itself as another very appealing approach that tightly couples communications and computations with the physical environment (now with location awareness).

This technique opens the possibility of deploying large-scale wireless sensor networks that can efficiently track how a physical quantity varies over space. The basis for the design of this approach is an interpolation scheme. This interpolation is a compact representation of sensor data at a given moment and it can be obtained efficiently.

The interpolation scheme can be summarized as follows. Each node is assumed to take sensor readings and to know its location. All nodes use the same function to interpolate the sensor data. Nodes start with the interpolation function being a flat surface. Then each node computes the error between its sensor reading and the interpolation function evaluated on its location. Exploiting the dominance MAC protocol, the nodes then use their error as part of the priority used to contend for the medium, such that the data point with the MAX of all errors is found. The data point found is then used by all nodes to recompute the interpolation function. Nodes iterate through this procedure for a predefined number of iterations (k). At the end of these iterations, the subset of k nodes that contribute to the interpolation is found. k is a parameter that defines the accuracy of the interpolation.

Interpolation Scheme Details

The approach for obtaining an interpolation of measurements more formally, using pseudo-code is now presented; this pseudo-code returns an upper bound on the error of the interpolation.

Let e_i denote the magnitude of the error at node N_i; that is:

$$e_i = |s_i - f(x_i, y_i)| \tag{5.3}$$

and let e denote the global error; that is:

$$e = \max_{i=1..m} e_i \tag{5.4}$$

The goal is to find $f(x,y)$ that minimizes e subject to the following constraints: (i) the time required for computing f at a specific point should be low; and (ii) the time required to obtain the function $f(x,y)$ from measurements should be low. The latter is motivated by the fact that it is interesting to track physical quantities that change quickly; it may be necessary to update the interpolation periodically in order to track, for example, how the concentration of hazardous gases move. For this reason, we propose using weighted-average interpolation (WAI) [112] (also used in [113, 114]). WAI is defined as follows:

$$f(x,y) = \begin{cases} 0 & \text{if } S = \emptyset; \\ s_i & \text{if } \exists N_i \in S\text{: } x_i = x \wedge y_i = y; \\ \frac{\sum_{i \in S} s_i \cdot w_i(x,y)}{\sum_{i \in S} w_i(x,y)} & \text{otherwise.} \end{cases} \tag{5.5}$$

where S is a set of nodes used for interpolation. The weights $w_i(x,y)$ are given by:

$$w_i(x,y) = \frac{1}{(x_i - x)^2 + (y_i - y)^2} \tag{5.6}$$

Intuitively, Equations 5.5 and 5.6 state that the interpolated value is a weighted average of all data points in S and the weight is the inverse of the square of the distance. There are many possible choices on how the weight should be computed as a function of distance; the option taken here was is intended to avoid calculations of square root in order to reduced the execution time on platforms that lack hardware support for floating point calculations. This is the case for typical sensor network platforms (for example, [107, 80]).

The original version [112] of weighted-average interpolation uses all available sensor measurements for interpolation. But this would imply that computing Equation 5.5

from sensor readings had a time-complexity of O(m). Fortunately, it is often the case [115] that sensor readings exhibit spatial locality; that is, nodes that are close in space give similar sensor readings. For this reason, the interpolation will offer a low error even if only a small number of carefully selected nodes are in S.

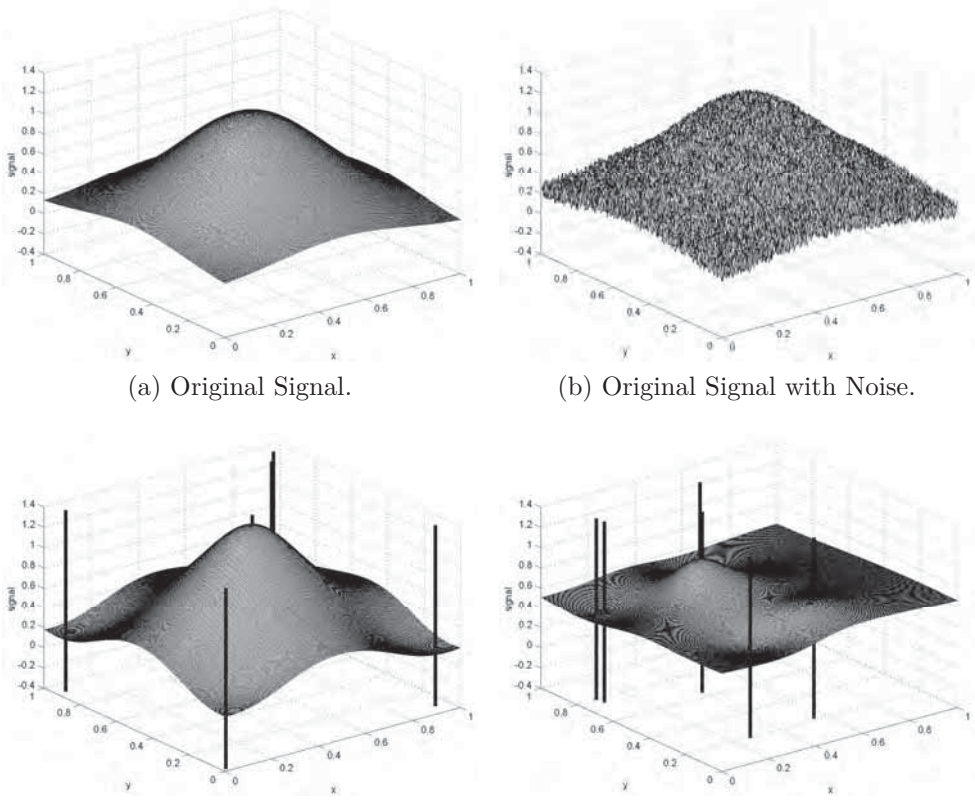
Hence, the goal is now to find those nodes that contribute to producing a low error in the interpolation as given by Equation 5.5. A number of k nodes that constitute the interpolation is selected, where k is a parameter of the algorithm that will control the accuracy of the interpolation. Recall that a prioritized MAC protocol can find the maximum among sensor readings. This feature can be exploited to find k nodes that offer a low value of the error. For this, the proposed distributed algorithm starts with an interpolation being a flat surface and then performs k iterations, where at each iteration the node with largest magnitude of the error between its sensor reading and the interpolated value will be the winner of the contention.

Algorithm 5.6 is designed based on this principle. It computes (on line 5) the error. This error is concatenated with the identifier of the node (together this forms

Algorithm 5.6 Finding a Subset of Nodes to be Used in WAI.

Require: All nodes start Algorithm 5.6 simultaneously.
Require: k denotes the number of interpolation points.
Require: A node N_i knows x_i,y_i and s_i.
Require: (MAXS+1) × (MAXNNODES+1) + MAXNNODES ≤ MAXP.

```
 1: function find_nodes() return a set of packets
 2:     S ← ∅
 3:     for j ← 1 to k do
 4:         myinterpolatedvalue ← f(x_i,y_i) in Equation 5.5
 5:         error ← abs( s_i - to_integer(myinterpolatedvalue) )
 6:         temp_prio ← error × (MAXNNODES + 1) + i
 7:         prio ← (MAXP+1) - temp_prio
 8:         snd_pack ←< s_i,x_i,y_i>
 9:         <winning_prio, rcv_pack> ← send_and_rcv( prio, snd_pack)
10:         S ← S ⋃ rcv_pack
11:     end for
12:     return S
13: end function
```



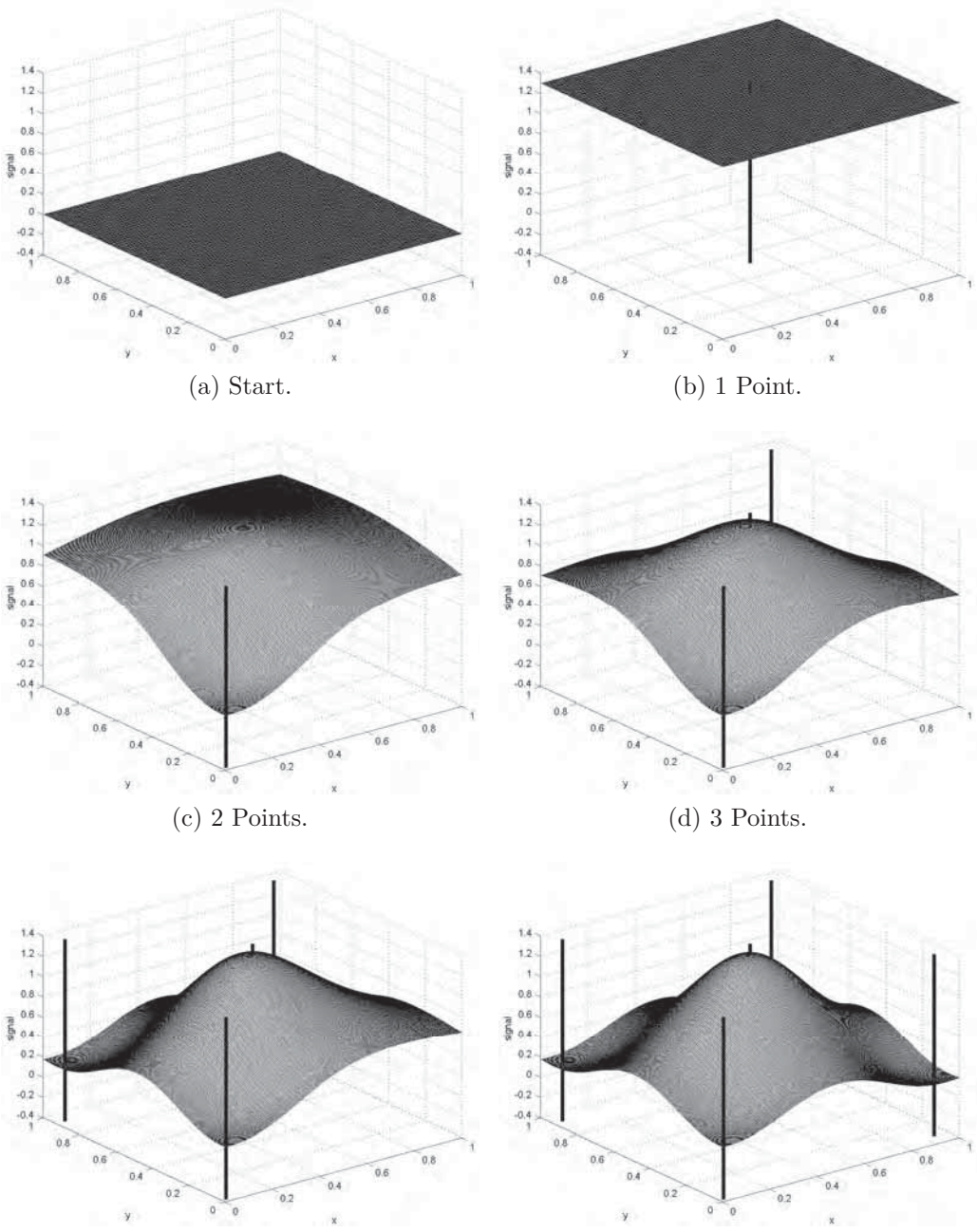
(a) Original Signal. (b) Original Signal with Noise.

(c) WAI w/ Carefully Selected Points. (d) WAI w/ Randomly Selected Points.

Figure 5.2: Interpolation Example 1.

the priority of the message) ensuring that all priorities are unique. All nodes send their messages in parallel (on line 9), being assured that only one node can win the contention for the medium and no collisions occur during the transmission of the location data.

The example in Figure 5.2 provides further intuition. In Figure 5.2a it is depicted the original signal that varies over space and the same signal with noise can be found in Figure 5.2b. The result of the interpolation given by our algorithm is shown in Figure 5.2c, where the location of the subset of $k = 5$ nodes that were selected to the interpolation is indicated with vertical lines. Finally, Figure 5.2d ilustrates an interpolation when 6 nodes are selected randomly. Clearly, performing weighted-average interpolation with randomly selected nodes (Figure 5.2d) gives poor interpolation.



(a) Start. (b) 1 Point.

(c) 2 Points. (d) 3 Points.

(e) 4 Points. (f) 5 Points.

Figure 5.3: Iterations Concerning Interpolation Example 1.

Figure 5.3 illustrates the operation of the interpolation scheme for $k = 5$, for the original signal in Figure 5.2a. At the beginning (a), no points are selected. Therefore,

the interpolation is a plane surface. Then (b), each node calculates the error between its sensor reading and the starting plane surface. This error is used in the contention of the MAC protocol, causing the node with the largest error to win, and thus, it is selected; (c)-(f) nodes proceed similarly, calculating their error to the currently interpolated surface and adding the node with the largest error to the interpolation, until k points have been selected.

The quality of the interpolation clearly depends on the characteristics of the signal. Our interpolation technique performs well as long as the signal does not change too abruptly (see Appendix A of [116]). This is often the case of signals that describe physical phenomena like temperature, light, or dispersion of a gas.

The interpolation technique presented was not designed to deal with sensor faults. However, mechanisms to deal with sensor faults can be introduced [117].

5.4 Algorithms for the MBD Case

It should be clear that the algorithms (presented in Section 5.3) for computing MIN, MAX and interpolation in a single broadcast domain do not work in a network with MBD. In this section, the algorithms will be extended.

In order to simplify the discussion, the focus will be given on the computation of MIN/MAX of sensor readings and later on theInterpolation case, which is a straightforward variation on the technique to obtain MIN for a SBD.

5.4.1 Computing MIN

In this section, it is assumed that time is slotted such that all nodes know the time when a timeslot begins, and they also know the identifier of the timeslot. One way to implement that is to use a platform that supports receiving an out-of band signal that provides a time reference for the timeslots. Such platform will actually be described in Section 6.3. Other examples can be found in the literature, such as the FireFly

sensor platform [80] that is equipped with an Amplitude Modulation (AM) receiver able to detect time-sync signals with a continental wide coverage. In this section, it is assumed that the duration of the timeslot is equal to the time it takes to run a tournament in the MAC protocol.

It is also assumed that all sensor nodes know when the computation should start, and do it periodically (for example, let all nodes start this computation at the beginning of a timeslot such that the identifier of the timeslot is divisible by 100). This is reasonable for applications that continuously detect an event. However in a multi-tiered architecture, where some nodes have a wider communication range, it is preferable to allow more high-powered sensor nodes to initiate a computation; this assumes that those high-powered sensor nodes have a communication range that covers the entire network.

The algorithm is composed of two main steps. At setup time, a topology discovery algorithm is executed to partition the network such that all nodes in each partition are in the same broadcast domain. Then, during runtime, nodes find the minimum sensor reading in all partitions and communicate these values to the leader.

Setup

The setup procedure must partition the network such that (i) each partition forms a single broadcast domain, (ii) a partition leader for each partition is selected, (iii) the partition leaders form a connected distributed set and (iv) to each partition is given a timeslot ensuring that no interfering partitions are active at the same time.

This procedure is started by selecting the partition leaders. To do this, a *Minimum Virtual Dominating Set* (MVDS) is selected as introduced in [21] (see Chapter 2 for additional reasoning). This algorithm approximates the solution for a MVDS(r) composed of the nodes colored black, where r is the virtual range used. It is important to note that, in this work, r is selected as a function of the communication range such that all nodes in each partition are in the same broadcast domain. Based on our

assumptions about the communication range, we can define $r = R_{co}/2$.

After running the propagation phase of the MVDS construction algorithm, the nodes selected as partition leaders report back to the leader the information about the topology of the network. This topology information is used by the leader to assign a timeslot to each partition such that the timeslot is unique from any 1 or 2-hop neighbors.

Runtime

At runtime, nodes have to find the minimum value within each partition, and then the partition leaders deliver these minimum values to the leader. Algorithm 5.7 provides the sequence of steps the nodes take during runtime.

Algorithm 5.7 Computing MIN in Multihop Networks

1: Each sensor node N_i takes a sensor reading. Let v_i denote this sensor reading.
2: Each node N_i in $PART_j$ waits until the time slot $SLOT(PART_j)$
3: Send an empty packet with the priority v_i.
4: After the tournament, $winnerprio_i$ denotes the minimum v_i
5: Communicate the results $winnerprio_i$ from partition leaders to the leader.
6: The leader takes the MIN of all $winnerprio_i$ that it receives.

In line 5 of Algorithm 5.7, while the minimum values are routed to the leader, partition leaders can perform in-network processing to the data received from other partitions.

A Running Example

To illustrate the algorithm operation a simple example is provided. Figure 5.4 shows network consisting of 100 nodes, where it is assumed that the algorithm is run after the sensor network is deployed (as described in Section 5.4.1). The algorithm partitions the network and selects the corresponding partition leaders. Figure 5.4 also depicts the partition leaders with a solid grey circle; the numbers in each node are the partition-IDs to which the node belongs (partition-IDs are assigned according to the partition leader address).

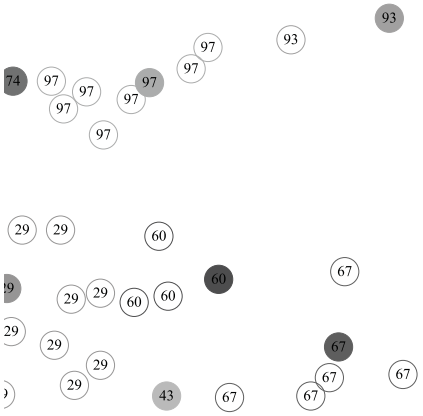


Figure 5.4: Partitioning and Partition Leaders for an Example Network.

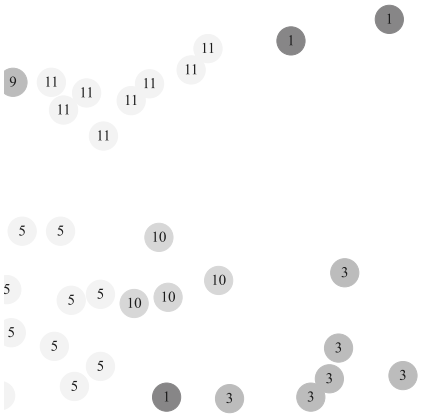


Figure 5.5: Timeslots Assigned to Partitions.

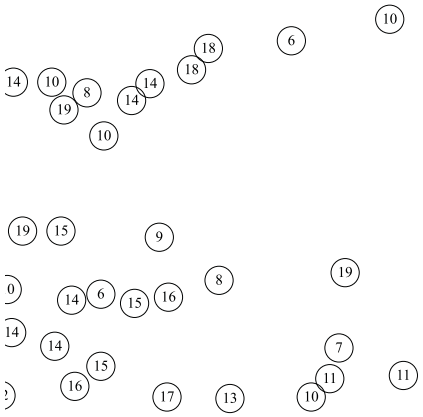


Figure 5.6: Each Sensor Node and the Original Sensor Reading.

Then, timeslots are assigned to each partition such that if two sensor nodes, in different partitions but in the same timeslot, broadcast simultaneously, then there is no collision. Figure 5.5 shows the timeslot assigned to each node. As illustrated, for this example, there will be eleven different timeslots.

Let us consider the algorithm that is executed at runtime. Figure 5.6 shows the temperature readings in all nodes. Nodes compete for the channel using their temperature readings as the priority and nodes do this in their assigned timeslot. After this competition, all nodes know the minimum of temperature in the partition. Figure 5.7 shows the result after the first timeslot. Observe that the nodes depicted in solid grey circles have all the same value within the corresponding partitions. This is because these nodes were assigned timeslot 1 and the values depicted are the minimum values in each partition, spread to all sensor nodes in the same partition. After 11 timeslots, all nodes have broadcasted their temperature reading. Figure 5.8 shows the result after the 11th timeslot. By then, every leader of a partition knows the minimum

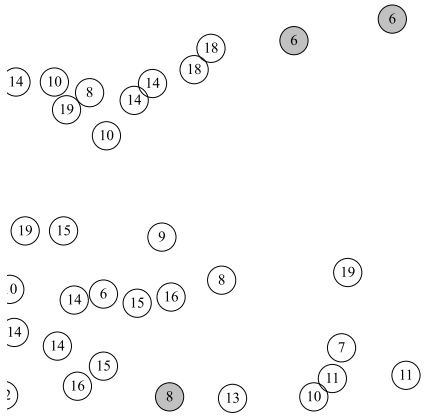


Figure 5.7: Result After Timeslot 1.

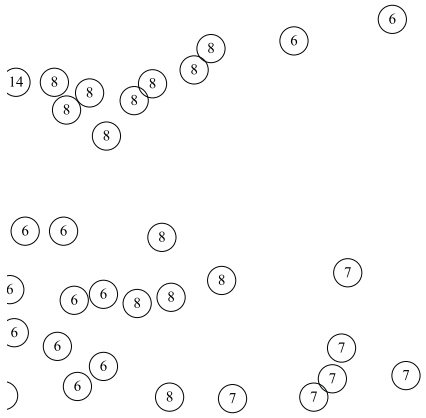


Figure 5.8: Result After Timeslot 11.

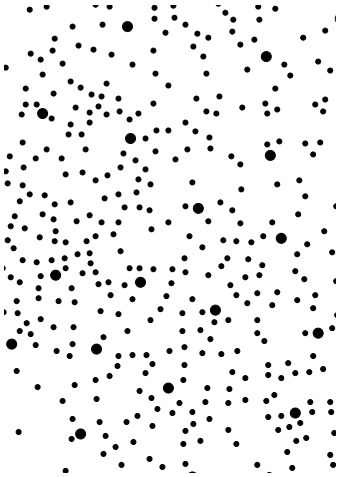


Figure 5.9: Large-scale Network Example.

temperature in the partition. Finally, nodes perform convergecast to the leader of the entire network. Observe that, due to the setup phase, nodes are organized in partitions where member nodes know their partition leaders and partition leaders known the other parent partition leaders who can forward message towards the leader node. Thus performing convergecast is trivial. After the convergecast, the leader knows that the minimum temperature in the entire network is 5.

To further illustrate why the algorithm is fast, a randomly generated network with 1000 nodes is depicted in Figure 5.9. In this figure, the 77 partition leaders are depicted with solid circles, slightly bigger than the other nodes. This example shows that our scheme scales well because only 17 unique timeslots are needed to obtain the

aggregated values from these 77 partitions. It can be observed than the increase in the number of required time slots is much smaller that the increase in the number of nodes.

So far we have assumed that all transceivers can only transmit in a pre-specified channel. But many wireless standards, such as IEEE 802.11, allow a transceiver to transmit on any channel. This feature can be used advantageously by assigning each partition its own channel (instead of assigning a timeslot to a partition) and this reduces the time required to perform step 2 in Algorithm 5.7.

As shown in Section 5.3.2, MAX can be computed similarly to MIN, using the bitwise negation of sensor readings.

5.4.2 Interpolation of Sensor Data

There are two ways of obtaining an interpolation of sensor readings in a multiple broadcast network. One approach is to create partitions (as described in Section 5.4.1), run the algorithm for obtaining the interpolation in each partition and then let each partition-leader communicate the data points, that constitute the interpolation in that partition, to the leader node of the network.

Another approach starts with a flat surface as an interpolation of the sensor signal in the area of the entire network. Then the node with the maximum error in the entire area is selected; this can be achieved by computing MAX in a multiple broadcast domain network as suggested in Section 5.4.1. When the leader node knows the node with the maximum error, it propagates this information to all nodes in the network (by sending this information to all partition leaders and then let each partition leader broadcast this information). In this way, all nodes in the entire network maintain a set of sensor readings on which they can compute a new interpolation. The procedure is then repeated k times.

5.5 Conclusions

In this chapter, it was shown how to use and take advantage of a dominance-based MAC protocol to efficiently compute aggregated quantities efficiently. The algorithms designed to exploit such MAC protocol have a time-complexity that is independent of the number of sensor nodes (for the SBD case, and increases very slowly for the MBD case). This is clearly important for WSN applications that operate under timeliness requirements, since faster computations may allow nodes to be awake for shorter periods (longer sleeping times), and thus energy consumption is also reduced, providing nodes a longer life-time.

The results in this chapter are considered to be relevant because: (i) several techniques are provided to perform highly scalable aggregate computations by exploiting a prioritized MAC protocol that supports a very large range of priority levels and is collision-free, assuming that priorities are unique; and (ii) a significant number of sensor networks are designed for large scale, dense networks and it is exactly for such scenarios that the algorithms presented excel.

Chapter 6

Improvements on the Implementation of WiDom

Contents

6.1 Introduction

In the previous chapter, it was shown that WiDom can be exploited to dramatically reduce the number of messages that need to be transmitted for obtaining certain aggregated quantities. This result can have a major impact only if the overhead introduced by running WiDom is low. Therefore, the following question persists:

Can aggregate quantities be efficiently obtained in wireless systems ?

The implementations of a dominance MAC protocol for wireless media previously presented (in Chapters 3 and 4) demonstrated that it is possible to have dominance/binary-countdown protocols in wireless systems. Nevertheless, its relatively high overhead may hinder its competitiveness against other approaches for data aggregation. The implementations used in those earlier chapters suffer from a significant overhead because of the time required to (i) perform carrier sensing and (ii) to switch between transmit and receive modes. A platform with better such characteristics has the potential to dramatically reduce the overhead and thereby render possible highly scalable aggregate computations for cyber-physical systems.

In this chapter, it is demonstrated that indeed it is possible to implement WiDom with a very low overhead and use that implementation for efficient distributed computations of aggregated quantities in cyber-physical systems. This is done by (i) describing the main design options of the new wireless hardware platform with appropriate characteristics for making dominance-based MAC protocols efficient; (ii) implementing dominance-based MAC protocols on this platform; (iii) implementing distributed algorithms for aggregate computations (MIN, MAX, Interpolation) as described in Chapter 5, using the new implementation of the dominance-based MAC protocol; and (iv) performing experiments to prove that such highly scalable aggregate computations in wireless networks are possible.

Additionally, in Section 6.4, is discussed and proposed another improvement related to providing better reliability in WiDom.

6.2 Impact of Hardware Shortcomings

Wireless dominance was successfully achieved in practice, as described in previous chapters. The implementations reported were based on off-the-shelf WSN platforms, with a radio transceiver that does not have favorable characteristics for the implementation of a wireless dominance protocol, and thus, these implementations exhibited a considerable overhead.

Specifically, the radio transceiver used in the previously reported implementations was the Chipcon CC2420 [108], a radio transceiver found in many WSN platforms. This transceiver, does not offer the most desirable characteristics for the implementation of dominance protocols. While the specific reasons may vary, this is unfortunately also true for a number of other radios currently used in WSN platforms.

First, WiDom requires that a carrier wave is transmitted for a short duration of time. While some radio transceivers allow to do this (e.g. the CC2420 radio transceiver), other radio transceivers only have a byte interface with the microprocessor, which limits the granularity of the duration for the transmission of carriers and introduces unnecessary overhead.

Second, WiDom requires that the radio is able to detect whether other nodes transmit a carrier wave. The ability to detect short pulses of carrier waves is instrumental for the development of an efficient dominance protocol. For example, the CC2420, can detect pulses of carrier waves, using its Clear Channel Assessment (CCA) functionality. The CCA functionality of the CC2420 radio computes the average Receiver Strength Indicator (RSSI) over the last 128 μs. To make a decision, this average is compared to a configurable threshold and then the CC2420 sets the CCA digital output pin accordingly. In practice, this means that T_{CS} will never be smaller than 128 μs. As seen in Section 3.4, carrier pulses were $T_{CS} = 486$ μs long in order to be reliably detected.

Finally, it is also necessary that the time to switch between transmission and reception is small. The CC2420, as an example, can take up to 192 μs to switch between these two modes, and then it needs another 128 μs (T_{RXTX} = 320 μs) until the first CCA operation can be made.

The combination of these factors results in wireless dominance that introduces a large overhead, and this limits the usefulness of such implementations in practice.

In the following section, the design of a platform that allows reducing the transmission/reception switching times and the time necessary for carrier sensing is briefly described. With this platform, it is possible to develop a competitive implementation of the WiDom protocol to enable efficient distributed computations of aggregated quantities in cyber-physical systems.

6.3 The Novel WiDom Platform

To address the problems described previously was developed a platform in the form of an add-on board that can be plugged into common WSN platforms such as the Mica family (Mica, Mica2 and MicaZ) and the CMU-FireFly [80]. This design allows to use other resources (hardware and software) that exist in those common platforms.

The following sections provide an overview of the platform and experiments performed. More details on the hardware development of this platform and experiments can be found in [118].

6.3.1 Overview

As mentioned earlier, the overhead of the contention of WiDom is dependent on: (i) the switching time between transmission and reception mode during the tournament; and (ii) the accuracy of the synchronization on the time when nodes should start the tournament.

A low switching time can be ensured by using two independent radio modules: one

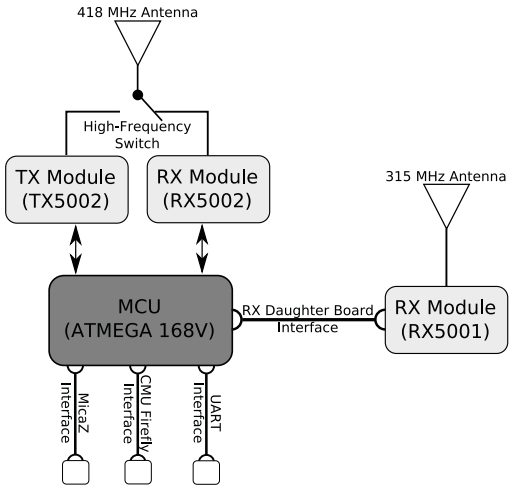
receiver and one transmitter. To allow the use of only one antenna, both modules share a common one, using a high-frequency switch.

To ensure good accuracy of the synchronization on the time when nodes should start the tournament. let a special node (called master node) send pulses periodically on a separate channel and when nodes receive a pulse they start executing the tournament. Each node (not the master node) has a separate receiver to detect those pulses. This brings the advantage that this receiver is always in reception mode so the synchronization is very accurate. This solution brings two additional advantages: (i) there is no need to wait for F time units (see the protocol automaton in Chapters 3 and 4); and (ii) the master node can transmit at high transmission power and use a frequency that gives a long range, making all nodes synchronized even in a multiple-broadcast network.

Our platform is comprised of a main board and a daughter board. They are constructed such that the main board is attached to a sensor node platform (Mica or FireFly) and the daughter board is attached to the main board. The main board sends and receives pulses in the tournament; the daughter board receives pulses on the separate channel and these pulses indicate the beginning of a tournament.

Figure 6.1a depicts a schematic of the main board and the daughter board. The platform includes interfaces to the Mica family and FireFly sensor platforms, an UART interface, for debugging purposes and an interface to the optional receive-only daughter board. Figure 6.1b shows the platform hardware in closer detail and Figure 6.1c depicts how the new boards stack on commonly available sensor platforms.

The correct operation of WiDom is very sensitive to timing; for example if the transmission of a carrier wave is performed $50\mu s$ later than it should, then the correctness property of WiDom (that the node with the lowest priority number wins) may be violated. For this reason, the main board is equipped with a microcontroller dedicated for running the WiDom protocol. It controls the radio modules and the high-frequency switch. And it receives commands from the host sensor node platform which priority it should compete on the channel with. It also communicates the priority of the winner



(a) Platform Schematic.

(b) Main Board and Rx Daughter Board.

(c) Platform Stacked on MicaZ and Firefly.

Figure 6.1: The Novel Hardware Platform.

to the sensor node platform. It is possible to transmit packets with the TX module on the main board but that is nore performed as more reliable packet transmission can typically be achieved with the transceiver on the sensor platform.

6.3.2 Achieving Reliable Tournaments

To achieve reliable tournaments, three main aspects had to be dealt with: symbol encoding, capture effect and bit stuffing. In wireless dominance, dominant bits are transmitted as a pulse of a carrier wave. To account for effects such as a node being able to detect a pulse from a node close by, but because it has adjusted its sensitivity for receiving from this node, it then cannot detect a pulse from a node far away, these

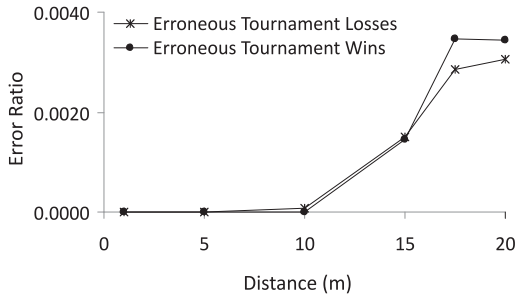


Figure 6.2: Failed Tournaments with Distance.

pulses must be transmitted with a period of silence between. This allows the receiver to adjust its sensitivity in order to detect carrier waves sent by distant nodes. Note that the switching time from transmit to receive mode is included in this period of silence. The main receiver module becomes unreliable when the medium is idle for a long period of time. This requires two modifications to the original WiDom protocol: (i) the first pulse after a long period of silence is composed of several pulses to adjust the receiver into an active state and (ii) bit stuffing must be introduced during the tournament, to avoid long periods of silence due to several consecutive recessive bits.

The length of the periods of silence between pulses and the amount of bit stuffing necessary is determined through experimentation and is described in Section 6.3.3.

6.3.3 Evaluation

The following characteristics of our platform deserve evaluation: (i) finding the minimum pulse duration and the transmission/reception switching time; (ii) assessing how the reliability changes with distance; and (iii) determining the power consumption.

To this end, several experiments were carried out. All experiments were conducted in an open-field environment, and all nodes were in non-obstructed line-of-sight.

First, to determine the minimum pulse duration and the transmission/reception switching time, two nodes were set at a distance d apart, and d was increased in steps of 1 *meter*, starting from 5 meters. For each step, the daughter board was

used to perform synchronization using an out-of-band signal; 10000 tournaments were performed using a static priority in each node. This procedure was then repeated for different combinations of pulse widths and intervals of silence. As a result of this experiment, a pulse width of $H = 40$ μs and an interval of silence between the bits $G = 50$ μs were selected as the best combination. Using these values, and considering bit stuffing, a tournament duration of $Q_{trnmt-SDB-new-platform} = 1754$ μs is obtained for 10 priority bits (this includes bit-stuffing).

To exhaustively test the reliability of the tournament at different distances, an experiment with 10 nodes was carried out. Furthermore, to test for cases where nodes can detect pulses from a node close by, but not from a node far away, the nodes were divided into two groups and each group of nodes placed at each end. The nodes in each group were placed with a minimum distance of 30 *cm* and the distance between the two groups of nodes varied from 1 to 20 meters, in steps of 5 meters. A table of random priorities was established and the winner of each tournament was computed offline. This table was then downloaded to the nodes and the priorities in the table were used in cycles until 150000 tournaments were performed. Because nodes know the priority of the winner in each tournament from the table computed offline, the number of tournaments failed (both events of failing to win or lose a tournament were counted) could be collected from each individual node. The results of this experiment are presented in Figure 6.2. It is possible to observe that, for networks where the distance between all nodes is below 5 meters, the communication can be considered error-free. Also, that the number of tournaments failed at a distance of 20 meters is very small.

The amount of energy consumed by the board was also studied. For this, the power consumed by a MicaZ platform with the board and the daughter board attached during a tournament was examined. During the measurements, the radio onboard the MicaZ was switched off, and the microcontroller was performing a busy loop. Because our platform needs to perform bit stuffing, each bit in our trace is composed of the actual bit (a pulse of the carrier wave, or monitor the medium) and an extra carrier pulse as

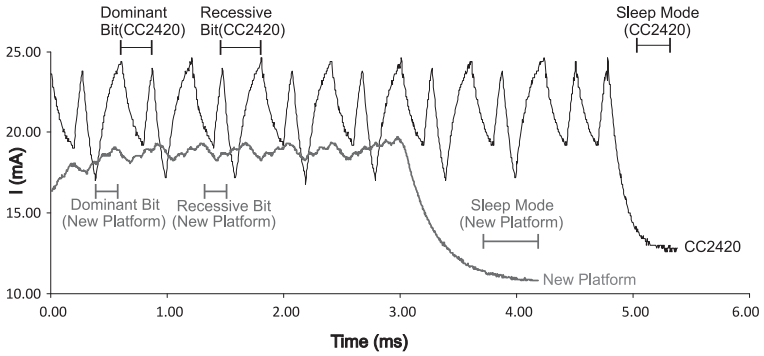


Figure 6.3: Trace of Power Consumption.

bit stuffing.

The amount of power that an implementation of wireless dominance using the CC2420 onboard the MicaZ to transmit a dominant and a recessive bit was also measured. For these measurements, a pulse width of $128 + 128$ μs was used. This duration corresponds to the minimum time needed to switch to transmit mode and perform one RSSI reading in the CC2420 (meaning that carrier sensing will never be faster, thus pulses should be, at least, this long). For the recessive bit, an interval of $192 + 128$ μs was considered, as this is the minimum time to switch to receive mode and again execute one RSSI reading. The microcontroller was also performing a busy loop during the measurements. Note that actually, a working implementation (such as the one in Section 3.4) would not be able to use pulses with such short duration to achieve reliable tournaments in a range of several meters.

Figure 6.3 presents both traces of power consumed (the supply voltage is 3V) for transmitting the same number (16) of priority bits in a tournament. One can observe that the new platform consumes significantly less energy than an implementation using the CC2420, even using the minimum time for the duration of the bits, as described above.

6.3.4 Comparative Analysis of the Time to Compute MIN

In this section, a brief analysis of the time to compute MIN using concrete technology is performed. This analysis allows to compare with the time to compute MIN using the algorithms in Chapter 5.

Let us consider the simple application scenario (as presented earlier in Chapter 1) where a node N_1 needs to know the MIN of the temperature readings among its neighbors. Let us assume that no other node attempts to access the medium before this node and that all nodes are within each others radio range. A naïve approach would imply that N_1 broadcasts a request to all its neighbors and then waits for the corresponding replies from them. As a simplification, assume that nodes have set up a scheme to orderly access the medium in a time division multiple access (TDMA) fashion, and that the initiator node knows the number of neighbor nodes. Then N_1 can compute a waiting timeout for replies based on this knowledge. Clearly, with this approach, the execution time depends on the number of neighbor nodes (m). In the following, this simple problem is considered and two solutions for it are compared. One using IEEE 802.15.4 and a solution based on a prioritized MAC protocol, as presented earlier.

More specifically, it is analyzed the time to compute MIN using a naïve algorithm, where all nodes are request to send their sensor readings. This analysis assumes that message transmission times dominate, and thus only considers the time needed to convey all messages necessary to compute MIN. For sake of simplicity, it is assumed that the number of nodes m is known, when using the naïve algorithm.

This comparison allows to have assess the time needed to perform such operations using concrete technology. The selection of IEEE 802.15.4 is due to the wide availability of compliant radio transceivers. Note that some of the parameters are derived from the transceiver's characteristics (such as tx/rx switching time or time for carrier sensing) and are not specific to the MAC protocol.

Non Beacon-enabled IEEE 802.15.4

For the implementation of our application using non beacon-enabled IEEE 802.15.4, it is assumed that a master node, the PAN coordinator, can poll all nodes sequentially. The transaction with each node involves two messages: a request and a reply. At the end of querying all m nodes, the PAN coordinator may trivially compute the MIN of all values. In such scenario, the time to compute MIN can be determined using the following simple expression:

$$T_{ComputeMIN-BE} = (T_{Request} + T_{Reply}) \times m \tag{6.1}$$

where $T_{Request}$ and T_{Reply} are the minimum times to send the request and reply packet, including the necessary backoffs, packet headers and footers, and interframe spacing (IFS). For this analysis, it is considered that there are no transmission retries (note this is a very optimistic assumption, as this implies that there are both no collisions and no transmission errors) and acknowledgments are disabled. The reasoning applied here is similar to the one found in [119, 120] when analyzing the maximum theoretical throughput of a non-beacon enabled IEEE 802.15.4. $T_{Request}$ and T_{Reply} are calculated based on a function $T_{MinPacket}(S)$ that provides the minimum time to send a packet with a payload size of S bits:

$$\begin{aligned} T_{MinPacket}(S) = \; & T_{InitialBackoff} \\ & + T_{MinPPDU}(S) + T_{Ack} + T_{IFS} \end{aligned} \tag{6.2}$$

where $T_{InitialBackoff}$ is the initial backoff period, which will be discussed later. The time to transmit the PHY protocol data unit (PPDU) with the minimum size allowed by the standard and a payload size of S bits is denoted by $T_{MinPPDU}(S)$. The time to transmit an acknowledgment is defined as $T_{Ack} = T_{AckPPDU} + T_{RxTx} = 544\ \mu s$ since it must include the time to send the acknowledgment packet ($T_{AckPPDU} = 352\ \mu s$ as defined in the standard [89]) and the time for the transceiver to switch from receive to transmit ($T_{RxTx} = 192\ \mu s$ is the maximum value defined in [89], and this is the value

found in common 802.15.4 transceivers, such as the CC2420 [108]). The interframe spacing, T_{IFS}, is set to the value of the minimum short IFS, 192 μs, because the size of the MAC protocol data unit (MPDU) to be sent is not above 18 bytes [89].

The time to transmit the PPDU with the minimum size allowed by the standard, with a payload of size S bits, can be defined as:

$$T_{MinPPDU}(S) = (S_{MinHeaders} + S + S_{Footer}) \times \tau_{bit} \tag{6.3}$$

where $S_{MinHeaders}$ is the sum of the sizes of the synchronization header (SHR), PHY header (PHR) and MAC header (MHR; from [89]: $S_{SHR} = 40$; $S_{PHR} = 8$; $S_{MHR} = 56$ bits, considering the minimum size of the addressing fields). The size of the MAC footer is $S_{Footer} = 16$ bits. The time to transmit one bit is $\tau_{bit} = 4\ \mu s$ (for a data rate of 250 kbps).

Calculating the Initial Backoff The IEEE 802.15.4 CSMA-CA algorithm dictates that nodes initially wait for a random number of backoff periods (the time of one backoff period is $BP = 320\ \mu s$ [89]) in the uniform interval $\left(0, 2^{BE} - 1\right)$, where BE is the backoff exponent, set according to parameter $macMinBE$, which can have an integer value in the interval $[0, macMaxBE]$, and $macMaxBE$ is an integer in the interval $[3, 8]$. The standard defines that, by default, $macMinBE = 3$. For analysing the minimum time to access the medium, there are two plausible choices for $macMinBE$: 0 or 1.

Considering that the collision avoidance mechanism for the first iteration of the IEEE 802.15.4 MAC algorithm is disabled ($macMinBE = 0$ [89]), then this would correspond to assigning $TInitialBackoff = 0$. This is however a very unrealistic assumption, since most practical setups cannot not justify disabling the first iteration of 802.15.4's MAC algorithm as this would result in many transmission retries (not considered here).

For the purpose of this analysis, it is considered the minimum value possible when

the collision avoidance mechanism for the first iteration of 802.15.4's MAC is enabled: $macMinBE = 1$. Thus, the mean number of initial backoff slots will be $(2^1 - 1)/2 = 0.5$, i.e. $T_{InitialBackoff} = 320 \times 0.5 = 160\ \mu s$. Note that, this value is very optimistic as retransmissions are not being considered (the probability of retransmissions is increased with a smaller initial backoff; in this analysis, the first transmission is always collision-free, and thus no retransmissions are made). This makes the time to compute MIN smaller than it would be in practice, but makes the analysis simpler. For our purposes, this will suffice.

Beacon-enabled IEEE 802.15.4 with Guaranteed Timeslots (GTS)

In the beacon-enabled mode of 802.15.4, beacon frames are periodically sent by the PAN coordinator to synchronize nodes that are associated with it. The Beacon Interval (BI) defines the time between two consecutive beacon frames. The beacon interval can be divided into an active period, and optionally an inactive period. The active period, called *Superframe*, is divided into 16 equally-sized timeslots, and its duration is defined by a parameter called Superframe Duration (SD). For this analysis, it is desired to have the smallest BI and SD, to impose the minimum protocol overhead possible. Given this constraint, $BI = SD = 15360\ \mu s$, which is the minimum SD defined by standard [89]. This means that nodes are at a 100% duty cycle and the duration of each timeslot is $T_{GTS} = SD/16 = 15360/16 = 960\ \mu s$. In IEEE 802.15.4, a Superframe is further divided into a contention access period (CAP), where a slotted version of CSMA-CA is employed and a contention-free period (CFP), composed of a maximum of 7 timeslots, named *guaranteed timeslots* (GTS). Furthermore, within the same Superframe, each node can only have one GTS upstream and one GTS downstream. For our application, only the CFP is used and it is assumed that, at each BI, the PAN coordinator assigns one timeslot to each node, such that all nodes are queried as soon as possible. Thus, $T_{ComputeMIN-GTS}$ is the time to send m messages in beacon-enabled IEEE 820.15.4 using GTS, and can be computed as follows:

$$T_{ComputeMIN-GTS} = \lfloor \frac{m}{7} \rfloor \times BI + (m \ mod \ 7 - 1) \times T_{GTS} + T_{Reply} + T_{MinCAPLen} \quad (6.4)$$

where the size of the CAP in the last Superframe necessary to contact all nodes is $TMinCAPLen = 7040\ \mu s$, the minimum defined in the standard. Note that this is a lower bound on the time to send m messages.

Idealized TDMA Using a IEEE 802.15.4-Compliant Transceiver

A comparison with an idealized TDMA mechanism is also made. The purpose here is to implement a MAC protocol with the lowest overhead possible, but still use realistic parameters from a IEEE 802.15.4 compliant transceiver.

To do this, it will be assumed that nodes are very accurately synchronized and a time slot has been assigned to each node, such that the nodes can orderly access the medium.

We can say that the time to send m messages in our idealized TDMA is:

$$T_{ComputeMIN-Ideal} = T_{ITS} \times m \quad (6.5)$$

where T_{ITS} is the length of a timeslot. Assuming that each timeslot has exactly the time needed for a reply from the node: $T_{ITS} = T_{Reply}$. T_{Reply} can be computed using Equation 6.2.

Exploiting a prioritized MAC

Let us now exploit a prioritized MAC protocol (such as WiDom) to compute MIN. Assume that the range of the analog to digital converters (ADC) on the sensor nodes is $[0, 2^{10}[$, and that the MAC protocol can, represent as many priority levels ($npriobits = 10$), and Algorithm 5.1 is used to compute MIN.

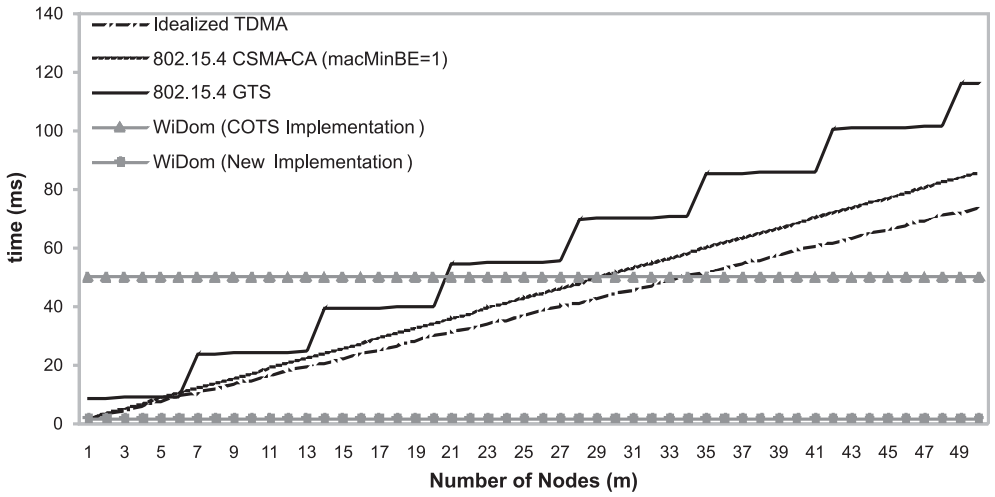


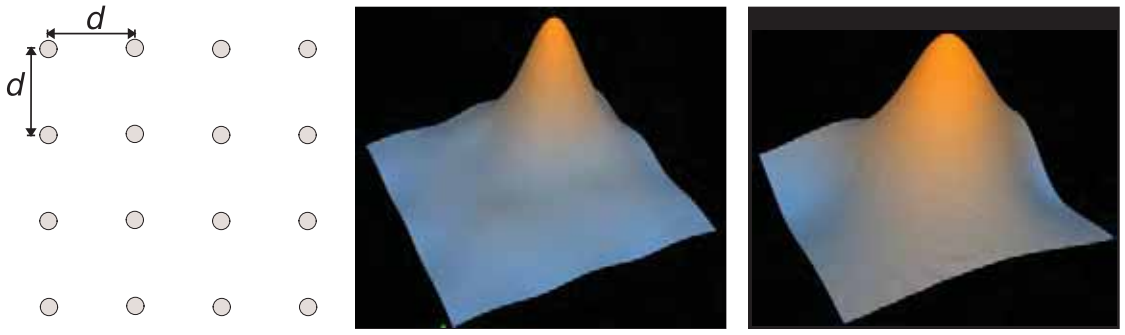
Figure 6.4: Time to Compute MIN in Function of m.

Note that, in this scenario, the time to compute MIN does not depend on m. The time to compute MIN is equal to the time to perform one arbitration of the prioritized MAC protocol. The implementation, described in Chapter 6, has shown that this protocol can be implemented, and the time to perform the arbitration is:

$$T_{ComputeMIN} = Q_{trnmt-SDB-new-platform} = 1754\ \mu s, fornpriobits = 10. \quad (6.6)$$

Computing MIN for m Nodes

The analysis presented in the previous sections can now be applied to evaluate the time to compute MIN in function of the number of nodes. Figure 6.4 plots the results for the IEEE 802.15.4 options described, the prototype implementation of WiDom in Chapter 3 and the implementation using specialized hardware in Chapter 6. For systems with $m \geq 34$, the implementation of WiDom (see Chapter 3) is superior than even the more idealistic setup using IEEE 802.15.4. Note that a more realistic comparison would be with the version based on GTS, and for this, the previous implementation of WiDom starts to be more advantageous for $m \geq 21$. The new implementation is always better for systems with $m \geq 2$.



(a) Node Placement Grid. (b) Original Signal. (c) Resulting Signal ($k = 5$).

Figure 6.5: Interpolation Experiment.

6.3.5 Demonstration of Interpolation

Using the novel implementation of WiDom, an experiment to study the reliability of the interpolation process was setup. The topology of the experiment includes 16 nodes (the number of prototype platforms available at this time) placed at equal distances d between each other, to form a 4×4 grid, as depicted in Figure 6.5a. A signal as depicted in Figure 6.5b was constructed and fixed the data points in each node according to it.

In this way, the resulting interpolation (depicted in Figure 6.5c) was known in advance. The interpolation was performed 10 000 times for each time the distance d was changed. The results are presented in Table 6.1. The experiments were conducted in an office environment.

It is possible to observe that even with such a demanding application (performing interpolation requires that not only the node with highest priority wins, but also that all other nodes perceive the right priority of the winner), our platform offers reliable

Table 6.1: Interpolation Experiment Results.

d (*meters*)	Error Ratio (%)
0.5	0.106
1	0.148
2	0.155

support for dominance-based aggregate computations.

6.4 Improving the Reliability of WiDom-SBD

In this section is discussed and proposed a simple modification to WiDom that improves its reliability. The idea to explore is to re-broadcast dominant bits; a concept that was already used in Chapter 4 to tackle the so-called hidden node problem.

6.4.1 Vulnerability

When the transceiver performs carrier sensing, it measures the amount of energy in the frequency band being used and compares this energy to a threshold. If the measured energy is above this threshold then it is perceived as an unmodulated carrier wave; otherwise it is considered as not being an unmodulated carrier wave. We assume that this threshold is set sufficiently high so that whenever no computer nodes send a packet, it holds that no transceiver declares that it has detected an unmodulated carrier. Note that frequently it is possible to trade effective communication range for a lower false positive probability, by raising the detection threshold.

Based on the above reasoning, let us briefly study the fault scenarios of WiDom. Consider the following four possible fault-scenarios: (i) an out-of-band signal was transmitted but there was a computer node which did not perceive it; (ii) no out-of-band signal was transmitted but there was a computer node which perceived an out-of-band signal; (iii) a carrier wave was transmitted by one of the computer nodes but there was another computer node which did not perceive this unmodulated carrier wave; and (iv) no carrier wave was transmitted but one computer node perceived an unmodulated carrier wave. Considering the previously stated reasoning, the fault scenarios (ii) and (iv) cannot occur. Let us assume that fault-scenario (i) occurs. Then, the consequence is that the computer node which did not perceive this unmodulated carrier wave will not participate in the contention resolution phase, and hence it holds that this node has

`winner = false`. If the network is large there will be nodes in every contention that act in this way. Consider for example 100 nodes and a probability of unheard carrier to be 10^{-4} (which is reasonable considering previous experiments in Section 3.4.4) then it follows that approximately 1 node every 100 tournaments will not participate in the contention. This behavior is undesirable because its effect is similar to priority-inversion in uniprocessor scheduling. But the network as a whole will continue to make progress in the sense that one node will transmit and this transmission will be collision-free. The scenario (iii) is even more adverse. Furthermore, as the number of nodes increase, it becomes increasingly more likely that at least one node does not detect a transmitted carrier.

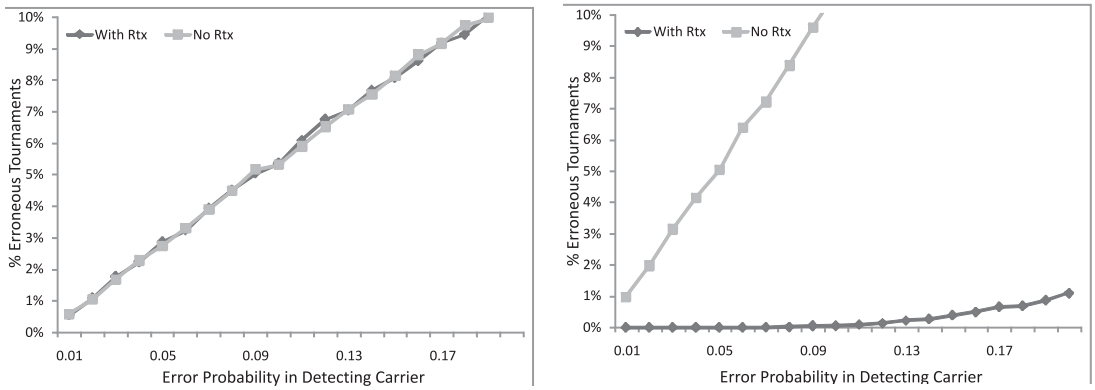
6.4.2 Protocol Modification

From the previous discussion, it follows that one important vulnerability of WiDom is the scenario where a node transmits a carrier wave and another node does not perceive this carrier wave. If the two nodes are closely located, then the signal strength from the transmitted carrier wave at the receiving node will be large and hence it is very unlikely that the receiving node will not perceive the carrier. It turns out that the technique used in Chapter 4 to propagate priority bits two hops away can be used to achieve this. This is done by performing the transmission of each bit in two stages. In the first stage, each node transmits its own priority bit. In the second stage, nodes retransmit the priority bit detected at the first stage (see Section 4.3.1 for details).

6.4.3 Evaluation

A simulator was implemented to experimentally test how the new protocol would perform under different carrier detection failure rates. Both the previous version of the protocol and the modified one were implemented, which also allowed comparing their performances.

The simulations were performed using 10 priority bits during the tournament. The



(a) n=2 (b) n=5

Figure 6.6: Probability of an Erroneous Tournament.

protocol was tested by varying the probabilities of missing the detection of a carrier pulse and the number of nodes. For each scenario, 10 independent simulation runs were executed. Each node was setup with one message stream having a unique priority and an exponentially distributed inter-arrival time, with an expected value ranging between 0.01s and 1s.

In all simulation runs nodes performed more than 10 000 tournaments. After each tournament, it was detected whether the correctness properties Collision-free, Progress and Prioritization were satisfied (see Section 3.3.2) for all nodes in the network. Tournaments where any node in the network failed to satisfy one of the properties are named erroneous tournaments. These erroneous tournaments were caused by failure to detect a priority bit. The number of erroneous tournaments observed was as given in the plots shown in Figure 6.6. In that figure, the version of WiDom without retransmission of priority bits is labeled as "no rtx"'.

The experiments show that without retransmission of priority bits, the number of failed tournaments increases very rapidly with the increase of carrier detection errors. The more nodes exist, more failed tournaments occur. Conversely, by increasing the number of nodes, it is clear that the protocol using retransmission performs markedly better. This is easily explained by the fact that, as more nodes exist in the network, it becomes more probable that a receiving node retransmits a dominant bit, previously

not detected. An experiment with ten nodes was also performed. In this experiment, no errors were found using the retransmission scheme. It can also be noticed that, for a network with two nodes, both versions of the protocol perform similarly. This is because, in this case, only one node is receiving at each time, thus no retransmissions occur.

6.5 Conclusions

The implementations of a dominance MAC protocol for wireless media previously presented (in Chapters 3 and 4) demonstrated that it is possible to have dominance/binary-countdown protocols in wireless systems. Nevertheless, its relatively high overhead may hinder its competitiveness against other approaches for data aggregation. The implementations used in those earlier chapter suffer from a significant overhead because of the time required to (i) perform carrier sensing; and (ii) to switch between transmit and receive modes. A platform iproving such characteristics has the potential to dramatically reduce the overhead and thereby render possible highly scalable aggregate computations for cyber-physical systems.

In this chapter, it was demonstrated it is possible to implement WiDom with a very low overhead and use that implementation for efficient distributed computations of aggregated quantities in cyber-physical systems.

Chapter 7

Discussion and Future Work

Contents

7.1 Introduction

This dissertation presented WiDom, a MAC protocol designed in close articulation with distributed algorithms for efficiently computing aggregated quantities in large-scale, dense sensor networks. This MAC protocol allows to efficiently obtain aggregate values such as minimum (MIN), maximum (MAX), number of nodes (COUNT), MEDIAN and interpolation in networked embedded systems. Another important characteristic of WiDom is that it is the first MAC protocol that enables static priority scheduling over wireless links.

This chapter concludes this dissertation. In Section 7.2, the results presented are reviewed and a discussion is provided on the contributions described in this dissertation. Section 7.3 discusses two scenarios assumed in this research work that are important to better comprehend the relevance of the contributions. Directions for future research are laid out in Section 7.4 and finally, Section 7.5, contains some closing considerations.

7.2 Review of Contributions

There are two important sets of contributions in this research: (i) the development and implementation of WiDom and the support for static priority scheduling over wireless links; and (ii) the algorithms for efficient data aggregation based on WiDom. Let us now review and briefly discuss each of them.

WiDom Development and Implementation. WiDom was proposed, a novel wireless MAC protocol inspired in dominance/binary countdown protocols which existed previously only for wired media [4]. This achievement is non-trivial. Firstly, implementations of dominance protocols for a wired medium are based on a wired–AND behavior of the bus, where the dominant signal overwrites the recessive signal. Secondly, these implementations require that nodes are able to monitor the medium while

transmitting. Clearly this does not easily extend to the case of wireless channels. Moreover, due to non-idealities of transceivers and the nature of the wireless medium, it was not obvious how a dominance protocol could be achieved.

WiDom supports a large number of priorities. Although this number of priorities introduces overhead, the application developer has the freedom to choose the number of priority levels required, and thus possibly reduce the overhead introduced. Nevertheless, such a large number of priorities can be supported by other prioritized protocols (see e.g., [77, 103]) but at the cost of an overhead several orders of magnitude higher.

The initial design of WiDom was created under the assumption of a SBD. An extension of WiDom for wireless networks with MBD was also developed. In such scenario, the hidden node problem must be dealt with. The proposed solution is the first prioritized and collision-free MAC protocol designed to successfully deal with hidden nodes without relying on out-of-band signaling.

The idea of retransmitting priority bits used to solve the hidden node problem can also be adapted to improve the reliability of WiDom in a SBD. In the presence of several nodes in the same broadcast domain, various nodes can cooperate in the transmission of the priority bits. This simple modification of the protocol can result in a substantial gain in the reliability, as the number of nodes increases.

WiDom was implemented and evaluated experimentally using Commercial-Off-The-Shelf (COTS) technology. The implementation of WiDom using COTS technology suffered however from a significant overhead. Therefore, a platform with better characteristics to implement dominance protocols was also studied and developed. This platform is the proof of concept that highly scalable aggregate computations in wireless networks are competitive in practice.

The experimental evaluation of WiDom shows that the probability that a message is transmitted collision-free, correctly prioritized and received (neither lost nor corrupted) by all other nodes is high and this reliability justifies the study of schedulability analysis techniques for sporadic messages in wireless networks; WiDom is an enabling technology allowing schedulability analysis (for example to exercise in practice

the analysis proposed by [64]) in wireless multihop networks with multiple broadcast domains. For the case of SBD, a response-time analysis for WiDom was developed and tested as well.

Efficient Data Aggregation. The research on efficient data aggregation in this dissertation is motivated by scenarios where even a small broadcast domain may contain several tens of sensor nodes. In these scenarios, the advantages of data aggregation solutions found in previous research are lost, since it is neither possible to apply data reduction functions to data coming from different sources nor is it possible to exploit the opportunities for parallel transmissions. In this thesis was demonstrated that it is possible to exploit a dominance-based MAC protocol to efficiently compute aggregate quantities with a time-complexity that is equivalent to the time of transmitting a single message, even if hundreds of nodes are in the same broadcast domain.

Concretely, the present work demonstrated that, in a single broadcast domain (SBD), the minimum value (MIN) can be obtained with a time-complexity that is $O(npriobits)$, where $npriobits$ is the number of bits used to represent the sensor data. In this case, the message complexity (and thus, the time-complexity) is independent of the number of sensor nodes. The same technique can be applied to obtain the maximum value (MAX).

Based on these techniques (of obtaining MIN or MAX), more elaborate aggregated quantities can be obtained. In this dissertation, useful examples such as the number of nodes (COUNT) and MEDIAN were also addressed.

Often, it is required to know how physical quantities (such as temperature) vary over an area. Clearly, the physical location of each node must be known then. For such systems, an algorithm that produces an interpolation of the sensor data as a function of space coordinates was proposed. The resulting interpolation is a compact representation of sensor data at a moment and is obtained efficiently.

The algorithms (to obtain aggregate quantities) were initially designed with the assumption of a SBD network. Nevertheless, in practice, most networks are not SBD

networks. Therefore, solutions for MBD networks were also studied and proposed. An algorithm for computing the MIN (or MAX) of sensor readings in a multihop network was proposed. That algorithm has the particularly interesting property of having a time-complexity that does not depend on the number of sensor nodes; only on the network diameter and the range of the value domain of sensor readings matter. Other more sophisticated algorithms were demonstrated also feasible for MBD networks.

These results are significant because often networks of nodes that take sensor readings are designed to be large scale, dense networks and it is exactly for such scenarios that the proposed algorithms (designed in close articulation with the MAC protocols) excel. The implementation of these algorithms in the hardware platform developed (in Chapter 6) shows that such highly scalable aggregate computations in wireless networks are indeed competitive in practice.

7.3 Discussion

There are two scenarios in this research work that are important to fully grasp the relevance of the contributions presented.

Dense Wireless Networks. The algorithms described in Chapter 5 are developed for large-scale, dense networks. In particular, it is in the presence of many nodes in the same broadcast domain that the advantages of an algorithm whose complexity does not depend on the number of nodes becomes evident. Note that, as show in Chapter 6, we can see advantages with as few as two.

Some may object accepting such scenario. Indeed, there are a few results which might advise against deploying dense networks. For instance, previous results on the capacity of ad-hoc wireless networks [121] show that, under certain assumptions, the capacity per node approaches zero as the number of nodes increases. However, several facts show that this is not the case in the context of WSN [122, 123], and other solutions can be devised (see, for example, the results reported in [124]).

One reason to search for other solutions is that data in sensor network is correlated, and this can be explored in several ways (e.g., [122, 124]). In this dissertation, the spatial correlation between sensor readings was exploited to perform a weighted-average interpolation and select only a subset of nodes (see Chapter 5).

A second reason differentiating WSN is that the communication pattern is often from several source nodes to a sink. This enables several techniques such as in-network data aggregation (a review of relevant works is provided in Chapter 2) or other techniques such as antenna sharing [123].

Another question is *why deploy a dense sensor network if we know that we will be gathering a lot of redundant data*? Deploying a dense sensor network might be convenient for several reasons. First, when considering the case where the individual cost of each node is negligible, then deploying redundant nodes might not be a primary factor in the cost of the system. Redundant nodes are useful for fault-tolerance (under certain fault assumptions) and noise immunity. Deploying redundant nodes also allows for a very fine spatial resolution in the sampling of the phenomena being observed. More importantly, deploying a dense network allows a better resolution of how the physical world is perceived; for example, when we are interested in high-resolution sampling in a certain region of interest, but we do not know in advance where that region is. This is expected to be essential in forthcoming innovative applications in cyber-physical systems.

Timeliness Guarantees in Wireless and Reliability. An important contribution of this research work is also a MAC protocol that supports static priority scheduling in wireless networks. This contribution encompasses its full impact, when assuming that *it is relevant to analyze the problem of providing timing requirements within a hard real-time context.*

It is often indicated that wireless links are unreliable, thus it is not meaningful to approach the problem from a guaranteed timeliness perspective.

It is true that designing a protocol with an upper bound on queuing time is not

sufficient to guarantee that hard real-time deadlines are satisfied in practice. However, it is a necessary step towards that goal. It is important to note that part of this problem is a technological one. The reliability of wireless networks has evolved noticeably over the last few years; it is safe to assume this evolution will continue. The experimental evaluation performed in this research work suggests that deadline misses due to noise are rare, so this provides evidence that it is still useful to consider hard real-time bounds on queueing delays. Obviously, any kind of guarantees are subject to some assumptions.

For the case of WiDom in SBD networks, there is a range of techniques originally developed for the CAN bus that can be applied, and this is a very interesting research path to explore. These techniques include the schedulability analysis as presented in Chapter 3, which are easily extendable to consider acknowledgements and retransmissions, and also other techniques such as stochastic approaches to model faults [125], for example. Moreover, the scheme to improve the reliability of WiDom in a SBD results in a substantial gain in the reliability, as the number of nodes increases. This fact alone can enable hard real-time deadlines to be satisfied with a very high probability in practice.

Even if not employing a guaranteed framework, being able to enforce strict priorities is useful in general. One recent example can be found in [103], where the authors built a streaming audio application employing a prioritized MAC protocol [78, 77]. It was found that, although the overhead of this MAC protocol is high, it still offers better throughput than normal CSMA/CA protocols because such prioritized MAC protocols eliminate the overhead of back-off after collisions. The scalability of the work reported in [103] is partially limited to the small number of priorities supported by the MAC protocol adopted and it is exactly in this point that WiDom stands out. Compared to that MAC protocol, WiDom offers a much higher number of priorities for a given similar overhead.

7.4 Future Work

There are several opportunities that can be identified for further research:

- development of other mechanisms to obtain other aggregate quantities or perform other distributed computations;
- enclosing in a query processing system;
- integration with other communication protocols;
- power management schemes/duty cycling;
- improvement of current implementations;
- further development of radio hardware;
- maximize the number of parallel transmissions of WiDom-MDB.

The following paragraphs briefly discuss each one of these possible future research topics.

As shown in Chapter 5, the fact that MIN can be obtained efficiently by exploiting WiDom can serve as a building block for other computations like COUNT, MEDIAN or Interpolation. It is possible to foresee that other computations may eventually be devised out of similar ideas and this is a research topic to be explored.

The computations enabled by WiDom could be encapsulated in a query processing system, similar to TinyDB [126]. The query processing would receive the query specifications and map these into primitives that exploit WiDom adequately. Eventually, queries that cannot be more efficiently carried out by exploiting WiDom could be mapped into other primitives that do not work by directly exploiting WiDom.

One interesting possibility is to integrate WiDom in other communication protocols. For example, a TDMA protocol can benefit from WiDom by allowing several nodes to share the same timeslot. Inside that timeslot, contention is resolved using WiDom. This combination would allow reducing the TDMA cycle. Another similar

example is to integrate WiDom in the IEEE 802.15.4 beacon-enabled mode, where one timeslot could be reserved for WiDom. In this case, during that timeslot, all nodes are allowed to access the medium using WiDom. Both examples can improve the schedulability of the system.

WiDom is energy efficient as it can avoid packet collisions and enable very efficient computations, but it lacks energy efficiency in the sense that it requires that nodes constantly monitor the medium. However, this does not need to be the case. WiDom is compatible with power management schemes proposed in WSN, such as coordinating activity/sleep schedules between the nodes (e.g. [127]). This is an important topic for further research.

At this point, the implementations of WiDom (and distributed algorithms) available are considered only as proof-of-concept prototypes. This means that the implementations were not developed for general use and they require some effort of understanding the details of the implementation in order to be used. This is not a topic for research, but it is important to consider this fact in order to asess the amount of effort needed to experiment with WiDom and/or further develop it.

The development of an efficient implementation of WiDom is still an important subject for future work. While the platform in Chapter 6 provides an indication that the overhead of the protocol can be low such that the algorithms based on exploiting WiDom can be very competitive, there is still a lot of work to be done in this aspect. The platform presented is a prototype developed from components commercially available at the time. The development and research of better radios for the execution of WiDom is a topic that would benefit the algorithms for obtaining aggregate quantities presented and would also enable low overhead static priority scheduling in wireless systems.

Finally, the proposed protocol for WiDom in a network with MBD (Chapter 4), does not maximize the number of parallel transmissions. This is a problem that can be difficult to solve, but, in practice, strategies as planning the priorities in the nodes such that multihop competing is avoided can provide interesting solutions.

7.5 Conclusions

This research work started by formulating the following hypothesis: *Is it possible to compute aggregate quantities with a time-complexity that is independent of the number of sensor nodes?*

To achieve this goal, WiDom was designed in close articulation with the data aggregation mechanisms. By exploiting the properties of a prioritized MAC protocol, the formulated hypothesis can be supported in SBD networks. Effectively, the time-complexity of the algorithms for obtaining aggregate quantities in SBD networks only depends on the sensor value range. In the case of a MBD network, the time-complexity of the algorithms developed also depends on the network diameter.

One important part of this research was dedicated to make this approach effective in wireless networks. This included the design and implementation of a prioritized MAC protocol in wireless media. Several implementations (in Chapters 3, 4 and 6) have shown that such approach is viable. While evidence was presented that the approach is feasible and the algorithms based on exploiting WiDom can be very competitive (see Chapter 6), there is a big gap to fulfill until such mechanisms are useful in more general settings. Filling this gap involves a wide range of aspects, from development of a framework to ease the use of such techniques by application developers, maturation of radio hardware or integration with other communication protocols.

This research work has demonstrated innovative mechanisms for obtaining certain aggregate quantities in dense wireless sensor networks. The work included the development of a prioritized MAC protocol that enables these mechanisms and can also efficiently schedule sporadic messages. Despite the dificulties yet to overcome, these constitute an attractive set of solutions for emerging Cyber Physical Systems.

Appendix A

List of Papers By the Author

This is a list of publications that reflects the results achieved during the development of the research work presented in this dissertation. Most of the results are included in this dissertation. There was some research that was not included in this thesis and is here listed for completeness.

Results Included in This Thesis:

1. B. Andersson, N. Pereira, and E. Tovar, "Using a Prioritized MAC Protocol to Efficiently Compute Aggregated Quantities," in *5th Intl Workshop on Real Time Networks (RTN'06)*, Dresden, Germany, 2006.

2. N. Pereira, B. Andersson, and E. Tovar, "Implementation of a Dominance Protocol for Wireless Medium Access," in *12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, Sydney, Australia, pp. 162-169, 2006.

3. N. Pereira, B. Andersson, and E. Tovar, "WiDom: A Dominance Protocol for Wireless Medium Access," *IEEE Transactions on Industrial Informatics,* vol. 3 (2), pp. 120-130, May 2007.

4. N. Pereira, B. Andersson, E. Tovar, and A. Rowe, "Static-Priority Scheduling over Wireless Networks with Multiple Broadcast Domains," in *28th IEEE Real-Time Systems Symposium (RTSS'07)*, Tucson, Arizona, USA, 2007, pp. 447-456.

5. B. Andersson, N. Pereira, W. Elmenreich, E. Tovar, F. Pacheco, and N. Cruz, "A Scalable and Efficient Approach for Obtaining Measurements in CAN-Based Control Systems," *IEEE Transactions on Industrial Informatics (TII),* vol. 4 (4), pp. 80-91, May 2008.

6. N. Pereira, B. Andersson, E. Tovar, and P. Carvalho, "Efficient Computation of Min and Max Sensor Values in Multihop Networks - by exploiting a prioritized MAC protocol", Springer Lecture Notes on Electrical Engineering (LNEE), vol. 38, Intelligent Technical Systems, pp. 233-246, 2009.

7. N. Pereira, R. Gomes, B. Andersson, and E. Tovar, "Efficient Aggregate Computations in Large-Scale Dense WSN," in *15th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'09)*, San Francisco, California, USA, pp. 317-326, 2009.

8. N. Pereira, B. Andersson, and P. Carvalho, "Improving the Reliability of WiDom in a Single Broadcast Domain," in *IEEE Symposium on Industrial Embedded Systems (SIES'09) - Work-in-Progress Session*, Lausanne, Switzerland, pp. 144-147, 2009.

Other Results not Included in This Thesis:

1. B. Andersson, N. Pereira, and E. Tovar, "Disseminating Data Using Broadcast when Topology is Unknown," in *26th IEEE Real-Time Systems Symposium (RTSS'05), Work-in-Progress Session*, pp. 61-64, 2005.

2. B. Andersson, E. Tovar, and N. Pereira, "Analysing TDMA with Slot Skipping," in *26th IEEE Real-time Systems Symposium (RTSS'05)*, Miami, FL, USA, 2005.

3. B. Andersson, N. Pereira, and E. Tovar, "Delay-Bounded Medium Access for Unidirectional Wireless Links," in *15th International Conference on Real-Time and Network Systems (RTNS'07), available online at* `http://rtns07.irisa.fr/fichiers/actes.pdf`, Nancy, France, pp. 205-214, 2007.

4. N. Pereira, B. Andersson, and E. Tovar, "Exact Analysis of TDMA with Slot Skipping," in *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'07)*, pp. 63-72, 2007.

5. B. Andersson, N. Pereira, and E. Tovar, "Analysing TDMA with Slot Skipping," *IEEE Transactions on Industrial Informatics,* vol. 4 (4), pp. 225-236, 2008.

Bibliography

[1] John A. Stankovic, Insup Lee, Aloysius Mok, and Raj Rajkumar. Opportunities and obligations for physical computing systems. *Computer*, 38(11):23–31, 2005. 3, 9

[2] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. *IEEE Pervasive Computing*, pages 59–69, January-March 2002. 3

[3] Bhaskar Krishnamachari, Deborah Estrin, and Stephen Wicker. Modelling data-centric routing in wireless sensor networks. In *21st Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM'02*, New York, USA, 2002. 3, 15

[4] Aloysius K. Mok and Steve Ward. Distributed broadcast channel access. *Computer Networks*, 3:327–335, 1979. 5, 28, 41, 42, 153

[5] CAN specification, ver. 2.0, 1991. Bosch GmbH, Stuttgart, Germany. 5, 28, 41

[6] N. Pereira, B. Andersson, and E. Tovar. Implementation of a dominance protocol for wireless medium access. In *12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, pages 162–169, Sydney, Australia, 2006. 10

[7] N. Pereira, B. Andersson, and E. Tovar. WiDom: A dominance protocol for wireless medium access. *IEEE Transactions on Industrial Informatics (TII)*, 3(2):120–130, May 2007. 10

[8] Nuno Pereira, Björn Andersson, Eduardo Tovar, and Anthony Rowe. Static-priority scheduling over wireless networks with multiple broadcast domains. In *28th IEEE Real-Time Systems Symposium (RTSS'07)*, pages 447–456, Tucson, Arizona, USA, 2007. 10, 97

[9] N. Pereira, B. Andersson, E. Tovar, and P. Carvalho. Improving the reliability of widom in a single broadcast domain. In *IEEE Symposium on Industrial Embedded Systems (SIES'09), Work-in-Progress Session*, pages 144–147, Lausanne, Switzerland, 2009. 10

[10] N. Pereira, R. Gomes, B. Andersson, and E. Tovar. Efficient aggregate computations in large-scale dense WSN. In *15th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'09)*, pages 317–326, San Francisco, California, USA, 2009. 10

[11] N. Pereira, B. Andersson, E. Tovar, and P. Carvalho. Efficient computation of min and max sensor values in multihop networks - by exploiting a prioritized MAC protocol. *Springer Lecture Notes on Electrical Engineering (LNEE)*, 38, Intelligent Technical Systems:233–246, 2009. 10

[12] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. In *5th symposium on Operating systems design and implementation (OSDI'02)*, pages 131 – 146, 2002. 16, 23, 24

[13] Ameer Ahmed Abbasi and Mohamed Younis. A survey on clustering algorithms for wireless sensor networks. *Computer Communications Journal*, 30(14-15):2826–2841, 2007. 18, 19

[14] Michele Mastrogiovanni and Alessandro Panconesi. Localized protocols for ad hoc clustering and backbone formation: A performance comparison. *IEEE Trans. Parallel Distrib. Syst.*, 17(4):292–306, 2006. 19

[15] Stephen Hedetniemi Teresa W. Haynes and Peter Slater. *Fundamentals of Domination in Graphs*. Pure and Applied Mathematics: A series of Monographs and Textbooks. CRC, 1998. 19, 20

[16] Pierluigi Crescenzi and Viggo Kann. A compendium of np optimization problems, online at: http://www.nada.kth.se/ viggo/wwwcompendium/node11.html. 20

[17] Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20:374–387, 1996. 20

[18] Bevan Das and Vaduvur Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *IEEE International Conference on Communications (ICC'97)*, pages 376–380, vol.1, Montreal, Canada, 1997. 20

[19] Jie Wu and Hailan Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications (DIALM'99)*, pages 7–14, New York, NY, USA, 1999. ACM. 20

[20] Ivan Stojmenovic, Mahtab Seddigh, and Jovisa Zunic. Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 13(1):14–25, 2002. 21

[21] B. Deb, S. Bhatnagar, and B. Nath. Multi-resolution state retrieval in sensor networks. In *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pages 19–29, 2003. 21, 120

[22] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 56–67, New York, NY, USA, 2000. ACM. 22

[23] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *33rd Hawaii International Conference on System Sciences (HICSS'00), Volume 8*, page 8020, Washington, DC, USA, 2000. IEEE Computer Society. 23

[24] S. Lindsey and C.S. Raghavendra. PEGASIS: Power-efficient gathering in sensor information systems. In *IEEE Aerospace Conference Proceedings*, volume 3, pages 1125–1130, 2002. 23

[25] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, 31(3):9–18, 2002. 23, 24

[26] Suman Nath, Phillip B. Gibbons, Srinivasan Seshan, and Zachary R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 250–262, New York, NY, USA, 2004. ACM. 23, 25

[27] Wu Jianping, M McDonald, M Brackstone, Li Yangying, and Guo Jingjun. Vehicle to vehicle communication based convoy driving and potential applications of gps. In *Proceedings of the 2nd International Workshop on Autonomous Decentralized Systems*, pages 212– 217, 2002. 24

[28] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 457, Washington, DC, USA, 2002. 24

[29] B. Krishnamachari, D. Estrin, and S. B. Wicker. The impact of data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, pages 575 – 578, 2002. 24

[30] Mihaela Enachescu, Ashish Goel, Ramesh Govindan, and Rajeev Motwani. Scale-free aggregation in sensor networks. *Theoretical Computer Science*, 344(1):15–29, 2005. 24

[31] T. Abdelzaher, T. He, and John A. Stankovic. Feedback control of data aggregation in sensor networks. In *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC'04)*, pages 1490–1495 Vol.2, 2004. 24

[32] Primoz Skraba, Qing Fang, An Nguyen, and Leonidas Guibas. Sweeps over wireless sensor networks. In *5th international conference on Information processing in sensor networks (IPSN'06)*, pages 143–151, New York, NY, USA, 2006. ACM. 25

[33] Kai-Wei Fan, Sha Liu, and Prasun Sinha. Structure-free data aggregation in sensor networks. *IEEE Transactions on Mobile Computing*, 6(8):929–942, 2007. 25

[34] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422, 2002. 25, 34

[35] Fen Zhao and Leonidas J. Guibas. *Wireless sensor networks : an information processing approach*. The Morgan Kaufmann Series in Networking. Elsevier, Amsterdam, 2004. 26

[36] K. Jamieson, H. Balakrishnan, and Y.C. Tay. Sift: a mac protocol for event-driven wireless sensor networks. In *Third European Workshop on Wireless Sensor Networks (EWSN)*, Zurich, Switzerland, 2006. 26

[37] Rahul Mangharam, Anthony Rowe, Raj Rajkumar, and Ryohei Suzuki. Voice over sensor networks. In *27th IEEE International Real-Time Systems Symposium (RTSS'06)*, pages 291–302, 2006. 26

[38] Rong Zheng, Lui Sha, and Wei Feng. MAC layer support for group communication in wireless sensor networks. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference (MASS'05)*, pages 8 pp.+, 2005. 26

[39] B. Andersson, N. Pereira, and E. Tovar. Using a prioritized mac protocol to efficiently compute aggregated quantities. In *5th Intl Workshop on Real Time Networks (RTN'06)*, Dresden, Germany, 2006. 26

[40] Gang Zhou, Tian He, Sudha Krishnamurthy, and John A. Stankovic. Models and solutions for radio irregularity in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 2(2):221 – 262, 2006. 26

[41] Victor Shnayder, Bor-rong Chen, Konrad Lorincz, Thaddeus R. F. Fulford-Jones, and Matt Welsh. Sensor networks for medical care. In *3rd international conference on Embedded networked sensor systems (SenSys'05)*, pages 314 – 314, Demonstration Abstracts, San Diego, California, USA, 2005. 27

[42] H. Kopetz and G. Grunsteidl. TTP-a protocol for fault-tolerant real-time systems. *IEEE Computer*, 27(1):14–24, 1994. 27

[43] N. Malcolm and W. Zhao. The timed-token protocol for real-time communications. *IEEE Computer*, 27(1):35–41, 1994. 27

[44] B. Andersson, , E. Tovar, and N. Pereira. Analysing tdma with slot skipping. In *26th IEEE Real-Time Systems Symposium (RTSS'05)*, pages 15–24, Miami, Florida, USA, 2005. 27

[45] M. Rahnema. Overview of the gsm system and protocol architecture. *IEEE Communications Magazine*, 31(4):92–100, 1993. 27

[46] Anthony Rowe, Rahul Mangharam, and Raj Rajkumar. RT-link: A time-synchronized link protocol for energy- constrained multi-hop wireless networks. In *3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON'06)*, pages 402–411, 2006. 27, 38

[47] M. Caccamo and L. Y. Zhang. An implicit prioritized access protocol for wireless sensor networks. In *23rd IEEE International Real-Time Systems Symposium (RTSS'02)*, pages 39–48, Austin, Texas, 2002. 27, 39

[48] N Abrahamson. The ALOHA system - another alternative for computer communications. In *1970 fall joint computer communications - AFIPS Conference Proceedings*, pages 281–285, Montvale, 1970. 28

[49] F.A. Tobagi and L. Kleinrock. Packet switching in radio channels: Part I carrier sense multipleaccess modes and their troughtputdelay characteristics. *IEEE Transactions on Communication*, 23(12):1400–1416, 1975. 28

[50] F.A. Tobagi and L. Kleinrock. Packet switching in radio channels: Part II the hidden terminal problem in carrier sense multipleaccess and the busytone solution. *IEEE Transactions on Communication*, 23(12):1417–1433, 1975. 28, 30, 62

[51] C.-S. Wu and V. O.K. Li. Receiver-initiated busy-tone multiple access in packet radio networks. In *ACM workshop on Frontiers in computer communications technology*, pages 336 – 342, Stowe, Vermont, United States, 1987. 30

[52] P. Karn. MACA - a new channel access method for packet radio. In *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pages 134–140. AARL, 1990. 30, 33

[53] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: a media access protocol for wireless lan's. In *Applications, Technologies, Architectures,and Protocols for Computer Communication*, pages 212 – 225, London, United Kingdom, 1994. 30, 33

[54] F. Talucci and M. Gerla. MACA-BI (MACA by invitation). a wireless MAC protocol for high speed ad hoc networking. In *IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC'97)*, pages 435–439, 1997. 30

[55] C. L. Fullmer and J. J. Garcia-Luna-Aceves. Solutions to hidden terminal problems in wireless networks. In *ACM SIGCOMM'97*, pages 39 – 49, Cannes, France, 1997. 31

[56] Z. J. Haas, J. Deng, and S. Tabrizi. Collision-free medium access control scheme for ad hoc networks. In *Proc. of IEEE Military Communications Conference (MILCOM'99)*, volume 1, pages 276–280, Atlantic City, NJ, USA, 1999. 31

[57] Z. J. Haas and J. Deng. Dual busy tone multiple access (DBTMA)a multiple access control scheme for ad hoc networks. *IEEE Transactions on Communications*, 50(6):975 – 985, 2002. 31, 33

[58] R. Garcés and J.J Garcia-Luna-Aceves. Floor acquisition multiple access with collision resolution. In *International Conference on Mobile Computing and Networking*, pages 187 – 197, Rye, New York, United States, 1996. 32

[59] R. Garcés and J. J. Garcia-Luna-Aceves. Collision avoidance and resolution multiple access with transmission queues. *Wireless Networks*, 5(2):95 – 109, 1999. 32

[60] Li B. Jiang and Soung C. Liew. Improving throughput and fairness by reducing exposed and hidden nodes in 802.11 networks. *IEEE Transactions on Mobile Computing*, 7(1):34–49, 2008. 32

[61] Shyamnath Gollakota and Dina Katabi. Zigzag decoding: combating hidden terminals in wireless networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):159–170, 2008. 32

[62] A. Acharya, A. Misra, and S. Bansal. MACAP : A mac for concurrent transmissions in multihop wireless networks. In *IEEE International Conference on Pervasive Computing and Communication*, pages 505–508, Los Alamitos, CA, 2003. 33

[63] Ragunathan Rajkumar. *Synchronization in real-time systems : a priority inheritance approach*. Kluwer Academic Publishers, Boston, 1991. 33

[64] T. F. Abdelzaher, S. Prabh, and R. Kiran. On real-time capacity limits of multihop wireless sensor networks. In *IEEE Real-Time Systems Symposium (RTSS'04)*, pages 359–370, Lisbon, Portugal, 2004. 34, 101, 155

[65] Ajay Chandra, V. Gummalla, and John O. Limb. Wireless medium access control protocols. *IEEE Communications Surveys & Tutorials*, 3(2):1–15, 2000. 34

[66] Sunil Kumar, Vineet S. Raghavan, and Jing Deng. Medium access control protocols for ad hoc wireless networks: A survey. *Ad Hoc Networks Journal*, 4:326–358, 2004. 34

[67] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: A survey. *Computer Networks*, 47(4):445–487, 2005. 34

[68] Koen Langendoen. Medium access control in wireless sensor networks. In H. Wu and Y. Pan, editors, *Medium Access Control in Wireless Networks, Volume II: Practice and Standards (to be published, available online at http://www.st.ewi.tudelft.nl/ koen/papers/mac4wsn.pdf)*. Nova Science Publishers, 2007. 34

[69] Kurtis Kredo II and Prasant Mohapatra. Medium access control in wireless sensor networks. *Computer Networks*, 51(4):961–994, 2007. 34

[70] Imad Aad and Claude Castelluccia. Differentiation mechanisms for IEEE 802.11. In *Infocom*, pages 209–218, 2001. 35

[71] Michael Barry, Andrew T. Campbell, and Veres Andras. Distributed control algorithms for service differentiation in wireless packet networks. In *Infocom*, 2001. 35

[72] Dr-Jiunn Deng and Chang Ruay-Shiung. A priority scheme for IEEE 802.11 DCF access method. *IEICE Transactions on Communication*, E82-B:96–102, 1999. 35

[73] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic real-time tasks. *Performance Evaluation, Elsevier Science*, 22(4):237–250, 1982. 35

[74] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed multihop scheduling and medium access with delay and throughput constraints. In *7th annual international conference on Mobile computing and networking (MOBICOM'01)*, pages 200–209, Rome, Italy, 2001. 35

[75] Rusty O. Baldwin, Nathaniel J. Davis, and Scott F. Midkiff. A realtime medium access control protocol for ad hoc wireless local area networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(2):20–27, 1999. 35

[76] Jang-Ping Sheu, Chi-Hsun Liu, Shih-Lin Wu, and Yu-Chee Tseng. A priority MAC protocol to support real-time traffic in ad hoc networks. *Wireless networks*, 10(1):61–69, 2004. 36

[77] João L. Sobrinho and A.S. Krishnakumar. Quality-of-service in ad hoc carrier sense multiple access networks. *IEEE Journal on Selected Areas of Comm.*, 17(8):1353–1368, 1999. 36, 154, 158

[78] João L. Sobrinho and A. S. Krishnakumar. Real-time traffic over the IEEE 802.11 medium access control layer. *Bell Labs Technical Journal*, 1(2):172–187, 1996. 36, 158

[79] Xue Yang and Nitin H. Vaidya. Priority scheduling in wireless ad hoc networks. In *MobiHoc'02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 71–79, New York, NY, USA, 2002. ACM. 36, 42

[80] R. Mangharam, A. Rowe, and R. Rajkumar. Firefly: a cross-layer platform for real-time embedded wireless networks. *Real-Time Syst.*, 37(3):183–231, 2007. 37, 63, 96, 115, 120, 133

[81] L. Girod J. Elson and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *5th symposium on Operating systems design and implementation (OSDI'02)*, pages 147 – 163, 2002. 37

[82] R. Kumar S. Ganeriwal and M. B. Srivastava. Timing-sync protocol for sensor networks. In *1st international conference on Embedded networked sensor systems (SenSys'03)*, pages 138 – 149, 2003. 37

[83] G. Simon M. Maróti, B. Kusy and Á. Lédeczi. The flooding time synchronization protocol. In *2nd international conference on Embedded networked sensor systems (SenSys'04)*, pages 39 – 49, 2004. 37

[84] A. Patel M. Welsh G. Werner-Allen, G. Tewari and R. Nagpal. Firefly-inspired sensor network synchronicity with realistic radio effects. In *3rd international conference on Embedded networked sensor systems (SenSys'05)*, pages 142 – 153, 2005. 37

[85] Venkatesh Rajendran, Katia Obraczka, and J.J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. In *1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*, pages 181–192, Los Angeles, California, USA, 2003. 38

[86] Anuj Puri Sinem Coleri and Pravin Varaiya. PEDAMACS: Power efficient and delay aware medium access protocol for sensor networks. *IEEE Transactions on Mobile Computing*, 5(7):920–930, 2006. 38

[87] Rahul Mangharam and Raj Rajkumar. MAX: A maximal transmission concurrency mac for wireless networks with regular structure. In *IEEE Third International Conference on Broadband Communications, Networks and Systems (BROADNETS'06)*, San Jose, CA, USA, 2006. 39

[88] Thomas Watteyne, Isabelle Augé-Blum, and Stéphane Ubéda. Dual-mode real-time mac protocol for wireless sensor networks: a validation/simulation approach. In *1st international conference on Integrated internet ad hoc and sensor networks InterSense'06*, page 2, New York, NY, USA, 2006. ACM. 39

[89] IEEE. IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 14.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area networks (LR-WPANs), October, 2003. 39, 140, 141, 142

[90] Zigbee alliance website, online at: http://www.zigbee.org/en/index.asp. 39

[91] Anis Koubâa, Mário Alves, and Eduardo Tovar. IEEE 802.15.4: a federating communication protocol for time-sensitive wireless sensor networks. chapter of the book "sensor networks and configurations: Fundamentals, techniques, platforms, and experiments", Springer-Verlag, Germany, Jan. 2007. 40

[92] Anis Koubâa, Mário Alves, and Eduardo Tovar. A comprehensive simulation study of slotted CSMACA for IEEE 802.15.4 wireless sensor networks. In *5th IEEE International Workshop on Factory Communication Systems (WFCS'06)*, Torino, Italy, 2006. 40

[93] Anis Koubâa, Mário Alves, and Eduardo Tovar. Improving the IEEE 802.15.4 slotted CSMA/CA MAC for time-critical events in wireless sensor networks. In *5th International Workshop on RealTime Networks (RTN'06)*, Dresden, Germany, 2006. 40

[94] Anis Koubâa, Mário Alves, and Eduardo Tovar. GTS allocation analysis in IEEE 802.15.4 for realtime wireless sensor networks. In *14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'06),*, Rhodes Island, Greece, 2006. 40

[95] Anis Koubâa, Mário Alves, and Eduardo Tovar. Energy/delay trade-off of the GTS allocation mechanism in IEEE 802.15.4 for wireless sensor networks. *International Journal of Communication Systems*, 20(7):791–808, 2007. 40

[96] Anis Koubâa, Mário Alves, and Eduardo Tovar. i-GAME: An implicit GTS allocation mechanism in IEEE 802.15.4. In *18th Euromicro Conference on Real-Time Systems (ECRTS'06)*, Dresden, Germany, 2006. 40

[97] Jaap C. Haartsen. The bluetooth radio system. *IEEE Personal Communications*, 7(1):28–36, 2000. 40

[98] Jan Beutel, Oliver Kasten, and Matthias Ringwald. BTnodes a distributed platform for sensor nodes. In *1st international conference on Embedded networked*

sensor systems (SenSys'03), pages 292 – 293, Poster Abstracts, Los Angeles, California, USA, 2003. 40

[99] K. Tindell, H. Hansson, and A. Wellings. Analysing real-time communications: controller area network (CAN). In *15th IEEE International Real-Time Systems Symposium (RTSS'94)*, pages 259–263, 1994. 42, 60

[100] K. W. Tindell, A. Burns, and A. J. Wellings. An extendible approach for analyzing fixed priority hard real-time tasks. *Springer Journal of Real-Time Systems*, 6(2):133–151, 1994. 42

[101] Robert I. Davis, Alan Burns, Reinder J. Bril, and Johan J. Lukkien. Controller area network (CAN) schedulability analysis: Refuted, revisited and revised. *Springer Journal of Real-Time Systems*, 35(3):239–272, 2007. 42, 59

[102] T. You, C.-H. Yeh, and H. S Hassanein. CSMA/IC: A new class of collision-free MAC protocols for ad hoc wireless networks. In *28th IEEE Int. Symp. on Comp. and Comm. (ISCC'03)*, pages 843–848, 2003. 42, 101

[103] B. D. Bui, R. Pellizzoni, M. Caccamo, C. F. Cheah, and A. Tzakis. Soft real-time chains for multi-hop wireless ad-hoc networks. In *13th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS'07)*, pages 69–80, 2007. 49, 73, 154, 158

[104] L. George, N. Rivierre, and M. Spuri. Preemptive and non-preemptive real-time uniprocessor scheduling. Technical report RR-2966, INRIA, online at: http://www.inria.fr/rrrt/rr-2966.html, September 1999. 59, 60

[105] Aloysius K. Mok. Fundamental design problems of distributed systems for the hard-real-time environment. Technical report, Thesis (Ph.D.), Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, Cambridge, MA, USA, 1983. 59

[106] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. J. Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. *Software Engineering Journal*, 8:284–292, 1993. 60

[107] Crossbow. MicaZ - wireless measurement system product datasheet. urlhttp://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf. 63, 64, 95, 115

[108] Chipcon. CC2420 datasheet. http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf. 63, 64, 66, 96, 132, 141

[109] TinyOS. TinyOS website. http://www.tinyos.net/. 64, 107

[110] CMU. FireFly and Nano-RK website: Carnegie-mellon university; realtime and multimedia systems lab. http://www.nano-rk.org/. 95

[111] B. Andersson, N. Pereira, and E. Tovar. Estimating the number of nodes in wireless sensor networks. In *IPP-HURRAY Technical Report - TR-060702*, 2006. http://www.hurray.isep.ipp.pt/widom/hurray-tr-060702.pdf. 109, 110, 111

[112] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517 – 524, 1968. 115

[113] R. Tynan, G.M.P. O´Hare, D. Marsh, and D. O´Kane. Interpolation for Wireless Sensor Network Coverage. In *Proceedings of the Second IEEE Workshop on Embedded Networked Sensors*, pages 123– 131, 2005. 115

[114] M. Sharifzadeh and C. Shahabi. Supporting spatial aggregation in sensor network databases. In *Proceedings of the 12th annual ACM international workshop on Geographic information*, pages 166 – 175, 2004. 115

[115] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *Proceedings of the Third International Conference on Information Processing in Sensor Networks (IPSN04)*, 2004. 116

[116] B. Andersson, N. Pereira, and E. Tovar. A scalable and efficient approach to obtain measurements in can-based control systems. Technical Report HURRAY-TR-061102, IPP-HURRAY! Research Group, Institute Polytechnic Porto, Porto, Portugal, 2006. 119

[117] B. Andersson, N. Pereira, and E. Tovar. How a cyber-physical system can efficiently obtain a snapshot of physical information even in the presence of sensor faults. In *Sixth Workshop on Intelligent Solutions in Embedded Systems (WISES'08)*, Regensburg, Germany, 2008. 119

[118] Ricardo Gomes. Efficient implementation of a dominance protocol for wireless medium access. Technical report, Thesis (MSc), Polytechnic Institute of Porto, Dept. of Electrical Engineering, Co-Advised by Nuno Pereira, Porto, Portugal, 2008. 133

[119] B. Latré, P. De Mil, I. Moerman, B. Dhoedt, P. Demeester, and N. Van Dierdonck. Throughput and delay analysis of unslotted IEEE 802.15.4. *JNW*, 1(1):20–28, 2006. 140

[120] *Measuring effective capacity of IEEE 802.15.4 beaconless mode*, volume 1, 2006. 140

[121] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000. 156

[122] Daniel Marco, Enrique J. Duarte-melo, Mingyan Liu, and David L. Neuhoff. On the many-to-one transport capacity of a dense wireless sensor network and the compressibility of its data. In *2nd International Workshop on Information Processing in Sensor Networks (IPSN'03), Lecture Notes in Computer Science*, pages 1–16, Palo Alto, CA, USA, 2003. 156, 157

[123] H.E. Gamal. On the transport capacity of the many-to-one dense wireless network. In *IEEE 58th Vehicular Technology Conference (VTC'03-Fall)*, pages 2881–2885, Vol.5, Oct. 2003. 156, 157

[124] Himanshu Gupta, Vishnu Navda, Samir Das, and Vishal Chowdhary. Efficient gathering of correlated data in sensor networks. In *6th ACM Intl. symposium on Mobile ad hoc networking and computing (MobiHoc'05)*, pages 402–413, 2005. 156, 157

[125] Ian Broster, Alan Burns, and Guillermo Rodríguez-Navas. Probabilistic analysis of can with faults. In *23rd IEEE International Real-Time Systems Symposium (RTSS'02)*, pages 269–278, Austin, TX ,USA, 2002. 158

[126] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, 2005. 159

[127] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 12(3):493–506, 2004. 160