Cross-layer energy optimisation of routing protocols in Wireless Sensor Networks

By
Szymon Fedor

Thesis directed by: Dr. Martin Collier

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

April 2008

School of Electronic Engineering
Dublin City University

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy *is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.*

Signed:

ID number: 54167868

Date: 25/01/2008

Abstract

Recent technological developments in embedded systems have led to the emergence of a new class of networks, known as Wireless Sensor Networks (WSNs), where individual nodes cooperate wirelessly with each other with the goal of sensing and interacting with the environment. Many routing protocols have been developed to meet the unique and challenging characteristics of WSNs (notably very limited power resources to sustain an expected lifetime of perhaps years, and the restricted computation, storage and communication capabilities of nodes that are nonetheless required to support large networks and diverse applications). No standards for routing have been developed yet for WSNs, nor has any protocol gained a dominant position among the research community. Routing has a significant influence on the overall WSN lifetime, and providing an energy efficient routing protocol remains an open problem. This thesis addresses the issue of designing WSN routing methods that feature energy efficiency.

A common time reference across nodes is required in most WSN applications. It is needed, for example, to time-stamp sensor samples and for duty cycling of nodes. Also many routing protocols require that nodes communicate according to some predefined schedule. However, independent distribution of the time information, without considering the routing algorithm schedule or network topology may lead to a failure of the synchronisation protocol. This was confirmed empirically, and was shown to result in loss of connectivity. This can be avoided by integrating the synchronisation service into the network layer with a so-called cross-layer approach. This approach introduces interactions between the layers of a conventional layered network stack, so that the routing layer may share information with other layers. I explore whether energy efficiency can be enhanced through the use of cross-layer optimisations and present three novel cross-layer routing algorithms. The first protocol, designed for hierarchical, cluster based networks and called CLEAR (Cross Layer Efficient Architecture for Routing), uses the routing algorithm to distribute time information which can be used for efficient duty cycling of nodes. The second method - called RISS (Routing Integrated Synchronization Service) - integrates time synchronization into the network layer and is designed to work well in flat, non-hierarchical network topologies. The third method - called SCALE (Smart Clustering Adapted LEACH) - addresses the influence of the intra-cluster topology on the energy dissipation of nodes. I

also investigate the impact of the hop distance on network lifetime and propose a method of determining the optimal location of the relay node (the node through which data is routed in a two-hop network). I also address the problem of predicting the transition region (the zone separating the region where all packets can be received and that where no data can be received) and I describe a way of preventing the forwarding of packets through relays belonging in this transition region.

I implemented and tested the performance of these solutions in simulations and also deployed these routing techniques on sensor nodes using TinyOS. I compared the average power consumption of the nodes and the precision of time synchronization with the corresponding parameters of a number of existing algorithms. All proposed schemes extend the network lifetime and due to their lightweight architecture they are very efficient on WSN nodes with constrained resources. Hence it is recommended that a cross-layer approach should be a feature of any routing algorithm for WSNs.

Acknowledgments

I am grateful to many people for help, both direct and indirect, in completing this thesis. I would never achieve that goal without the support and suggestions of many colleagues, friends and family.

First of all I would like to thank my supervisor Dr. Martin Collier for continuous help, support and encouragement.

I wish to thank Prof. Barry McMullin, Dr. Stephen Daniels and Prof. Cormac Sreenan for serving on my doctoral committee and for their comments and suggestions.

An important source of inspiration and knowledge have been my colleagues from the *Dublin City University* an especially from the *Switching and Systems Laboratory*. I am particularly grateful to Damien O'Rourke for his valuable comments and suggestions. Our long discussions and collaboration on different tasks (especially long lasting experiments) were an unforgettable experience.

Support and help from my brothers, Marcin and Krzysztof, are also priceless. They were both very supportive emotionally to me during their numerous visits to Ireland. Also Marcin helped me to decide to start this PhD project.

The most important support I have received from my beloved wife Agnieszka. She encouraged me during difficult times and she has always believed in me. I want also to mention the immense emotional support I have got from our son Dawid.

Last but not least, I thank to my parents and to my parents-in-law for unconditional support and encouragement to pursue my interests. I always remember that I can count on my parents and they were very supportive to me during many difficulties. Also my mother-in-law helped me a lot during the final stage of the thesis. Dziękuję rodzicom oraz teściom za bezwarunkową pomoc i wsparcie w realizacji moich zainteresowań. Zawsze pamiętam że mogę liczyć na rodziców i często służyli mi pomocą w przezwyciężeniu różnych trudności. Również moja teściowa wiele mi pomogła w ostatniej fazie pracy doktorskiej.

My apologies if I have inadvertently omitted anyone to whom acknowledgment is due.

Dublin, Ireland, November 2008 *Szymon Fedor*

List of Publications

- S. Fedor and M. Collier, *Synchronisation service integrated into routing layer in Wireless Sensor Networks*, in Proceedings of IEEE Wireless Communications & Networking Conference (WCNC'08), Las Vegas, USA, April 2008.
- D. O'Rourke, S. Fedor, C. Brennan, and M. Collier, *Reception region characterisation using 2.4 GHz Direct Sequence Spread Spectrum radio*, in Proceedings of 4th workshop on Embedded networked sensors (EmNets'07), Cork, Ireland, June 2007.
- S. Fedor and M. Collier, *On the problem of energy efficiency of multi-hop vs one-hop routing in wireless sensor networks*, in Proceedings of 21st IEEE 21st Int'l Conference on Advanced Information Networking and Applications Workshops (AINAW '07), Niagara Falls, Canada, May 2007.
- S. Fedor, D. O'Rourke and M. Collier, *Cross-Layer Routing with Data Delivery Guarantee in Wireless Sensor Networks*, in Proceedings of ACM workshop on real-world Wireless Sensor Networks (REALWSN'05) in conjunction with ACM MobiSys, Uppsala, Sweden, June 2006.
- S. Fedor, *Promising future of Wireless Sensor Networks*, in InfoTel, vol. 9, Feb. 2005.
- S. Fedor and M. Collier, *An Intra-cluster Architecture to Prolong Wireless Sensor Network Lifetime*, in Proceedings of IEEE Conference on Software, Telecommunications and Computer Networks (SoftCOM'05), Split, Croatia, Sept. 2005.

CONTENTS

LIST OF FIGURES

PSEUDO-CODE LISTINGS

LIST OF TABLES

LIST OF ACRONYMS

AODV Ad hoc On-demand Distance Vector Routing

ARPA Advanced Research Projects Agency

ATM Asynchronous Transfer Mode

BER Bit Error Rate

BGP Border Gateway Protocol

CADR Constrained Anisotropic Diffusion Routing

CLEAR Cross Layer Efficient Architecture for Routing

CPU Central Processing Unit

DAG Directed Acyclic Graph

DAR Dynamic Alternative Routing

DCR Dynamically Controlled Routing

DNHR Dynamic Nonhierarchical Routing

DSDV Destination Sequence Distance Vector

DSR Dynamic Source Routing

DSSS Direct Sequence Spread Spectrum

DV Distance Vector

EGP Exterior Gateway Protocol

FSK Frequency Shift Keying

FSR Fisheye State Routing

FTSP Flooding Time Synchronisation Protocol

GAF Geographic Adaptive Fidelity

GEAR Geographic and Energy Aware Routing

GPSR Greedy Perimeter Stateless Routing

HWMP Hybrid wireless mesh protocol

IDRP Inter-Domain Routing Protocol

IERP Inter-zone Routing Protocol

IISP Interim Interswitch Signalling Protocol

ISDN Integrated Services Digital Network

IS-IS Intermediate System to Intermediate System

LAN Local Area Network

LEACH Low-Energy Adaptive Clustering Hierarchy

LS Link State

MANET Mobile Ad-hoc Network

MCFA Minimum Cost Forwarding Algorithm

MECN Minimum Energy Communication Network

MER Minimum Energy Routing

MNL Maximum Network Lifetime

MPR multipoint relays

MRLQSR Multiradio Link Quality Source Routing

MTE Minimum Transmission Energy

NLRI Network-Layer Reachability Information

NPDU Network Protocol Data Unit

OLSR Optimised Link State Routing Protocol

O-QPSK Offset Quadrature Phase-shift Keying

OSI Open System Interconnect

OSPF Open Shortest Path First

PEGASIS Power-Efficient Gathering in Sensor Information Systems

PNNI Private Network-Node Interface

PRR Packet Reception Rate

QoS Quality of Service

RA-OLSR Radio Aware Optimised Link State Routing

RIP Routing Information Protocol

RISS Routing Integrated Synchronisation Service

RSSI Received Signal Strength Indication

SAR Sequential Assignment Routing

SCALE Smart Clustering Adapted LEACH

SFD Start of Frame Delimiter

SNR Signal to Noise Ratio

SPIN Sensor Protocol for Information via Negotiation

TEEN Threshold-Sensitive Energy Efficient Sensor Network Protocol

TORA Temporally Ordered Routing Algorithm

TTL Time To Live

WMN Wireless Mesh Network

WRP Wireless Routing Protocol

WSN Wireless Sensor Network

ZRP Zone Routing Protocol

CHAPTER 1

Introduction

Embedded systems[1] have been an integral part of our daily life for decades. They can be used to control a washing machine, a cell phone, an engine of a car, or a treadmill in a gym. 98% of all computing devices are used in an embedded context [17] and some reports expect that the demand for embedded CPUs is 10 times as large as for general purpose PC CPUs [147]. The significant growth in the number of embedded shipments is expected to continue over the coming years [4].

Despite their maturity, embedded systems have attracted significant research attention in the last ten years, since technological developments and many decades of miniaturisation (following Moore's law [106]) have enabled the development of low power systems capable of performing various tasks, from sensing, data processing and storing to packet transmitting and receiving.

Hence it is possible to enhance the capabilities of embedded systems by interconnecting individual devices. As unconnected entities they still have very limited processing and storage resources in order to enable long operational time

[1]An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, sometimes with real-time computing constraints [11].

with limited battery resources. However, the interaction of these objects enables a lot of very different real-world applications like disaster relief (*e.g.* triage), environmental control and biodiversity mapping[2], or intelligent buildings.

A new class of networks has appeared in the last few years: the so-called Wireless Sensor Network (WSN). A WSN is a set of small autonomous systems, called sensor nodes (also known as motes), which communicate wirelessly and cooperate to solve at least one common application. These nodes have to collaborate to fulfill their tasks as, usually, a single node is incapable of doing so. Their task includes some kind of perception and conceivably a control of physical parameters. Each of these scattered sensor nodes has the capability to collect and route data either to other sensors or back to a base station (see Figure 1.1).

A WSN may potentially be deployed over a wide area with a distance of many kilometers separating the edge nodes. Because the energy resources of sensor nodes are very limited and transmission over long hops is very energy demanding, multi-hopping is required for most WSN applications as a means to extend network coverage and lifetime. Multi-hopping is also used to combat limited bandwidth and to reduce interference. Therefore, intensive research has been carried out in recent years in the area of WSN routing with the aim of developing efficient protocols which route data in this energy constrained, multi-hop environment.

1.1 Motivation

Routing in WSNs is very challenging due to the inherent characteristics that distinguish these networks from other wireless networks like mobile ad hoc networks or cellular networks. These include types of traffic flow (of sensed data from multiple sources to a particular base station), limited energy resources, lim-

[2]Biodiversity mapping consists of mapping and predicting the distribution of plants and animal species in a given habitat.

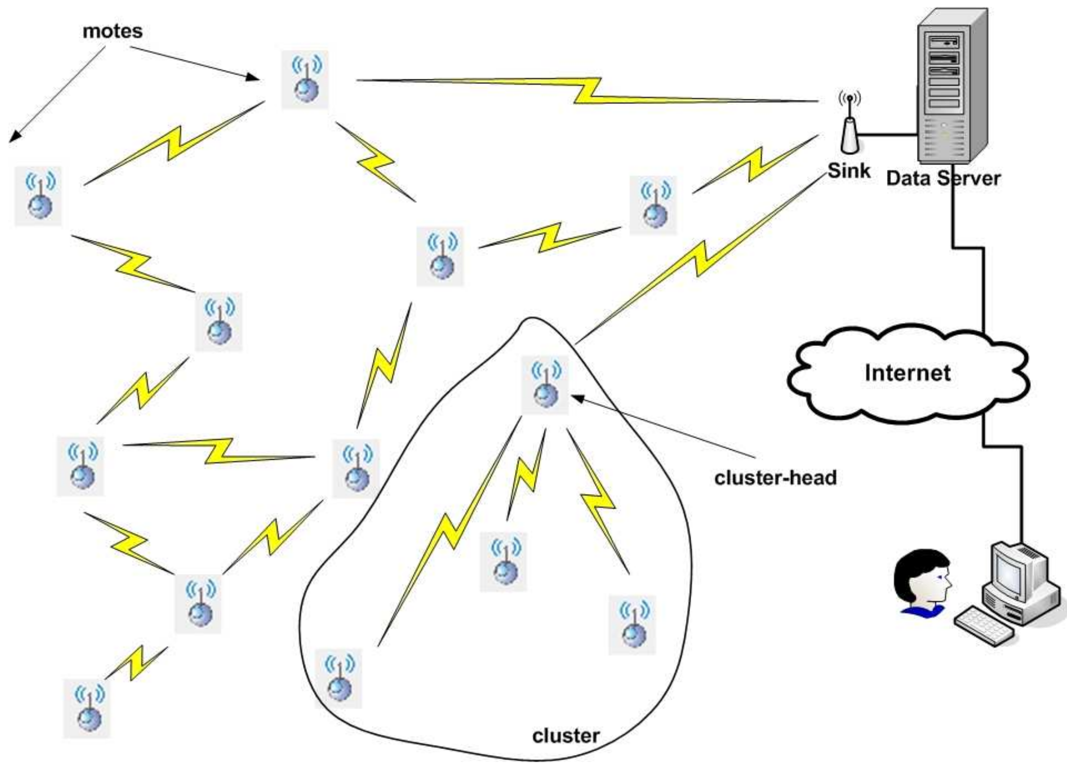


Figure 1.1: *Diagram of an illustrative WSN. Sensor nodes (motes) sense the environment and send measured samples to the base station (also called the sink or gateway). Neighbour sensor nodes may form a cluster and transmit their measurements to the cluster-head which is responsible for forwarding these packets to the sink. The base station collects and stores measured data into an integrated (i.e.) or external data server. A user may then access and analyse recorded measurements and also manage the WSN.*

ited processing and storage capacities of nodes, the redundancy of the transmitted data, and the large number of nodes which raises many challenges related to scaling (*e.g.* time-efficient packet delivery, interference, or large global addressing scheme).

Due to such differences, many new algorithms have been proposed for the routing problem in WSNs. These routing mechanisms have taken into consideration the inherent features of WSNs along with the application and architecture requirements. The task of finding and maintaining routes in WSNs is nontrivial since energy restrictions and sudden changes in node status (e.g., failure) may cause frequent and unpredictable topological changes. To minimise energy consumption, routing techniques proposed in the literature for WSNs employ some well-known routing tactics as well as tactics specific to WSNs, such as data aggregation and in-network processing, clustering, role assignment in function of node's abilities, and data-centric methods.

One technique to improve the efficiency of routing protocol is called cross-layer design where the traditional layered protocol architecture is violated by cutting across traditional layer boundaries with the purpose of performance optimisation, resource preservation or error resilience. Such techniques offer numerous possibilities for routing optimisation but they can also cause various problems. These and other related to the optimisation of routing protocols in WSNs aspects are considered in this thesis.

1.2 Thesis contribution

1.2.1 Problem statement

I concentrate in this thesis on routing methods for Wireless Sensor Networks. The main focus of this work is threefold:

- to identify cross-layer routing optimisation approaches incorporating time information;

- to investigate the transition region problem which often results in a poor routing performance and to propose adequate solutions;
- to analyse how information about network topology can enhance routing performance and to propose feasible techniques of routing optimisation using this information.

1.2.2 Summary of contributions

The main contributions presented in this thesis are:

- Cross-layer methods to integrate time synchronisation into the routing layer are proposed. The techniques I propose, Routing Integrated Synchronisation Service (RISS) and Cross Layer Efficient Architecture for Routing (CLEAR), offer excellent time synchronisation precision and an improvement of network lifetime for various WSN topologies.
- A cross-layer method, called Smart Clustering Adapted LEACH (SCALE), which enhances the energy efficiency of routing by incorporating topology information into the network layer is described. I have studied the influence of intra-cluster topology on the energy dissipation of the nodes and the results of my investigations are included in the SCALE method.
- A detailed study of the impact of hop distance on the network lifetime is provided. I determine when multi-hop routing is to be preferred over a direct transmission to the base station. I also describe conditions for which a two-hop strategy is optimal and in relation to it what is the optimal location of the relay node (the node through which data is routed in a two-hop network).
- A method for delimiting the transition region in an O-QPSK 2.4GHz WSN architecture is defined. I have studied factors influencing the extent and

location of the transition region and confirmed the theoretical results of my analysis with experiments in different environments. As a consequence I propose a method of routing optimisation that eliminates routes via nodes in the transition region.

1.3 Thesis outline

The reminder of this thesis is organised as follows:

In **Chapter 2** I define WSN routing and I identify challenges of designing routing protocol for WSNs. I present a survey of WSN routing algorithms also giving a brief overview of the historical deployment of standard routing protocols. I give examples of protocols which influenced the development of routing schemes for WSNs. This is followed by a description of the cross-layer design approach, its motivations, drawbacks and examples of implementation of various cross-layer architectures. Finally I review topics related to the research presented in this thesis: the wireless propagation model, the transition region, and time synchronisation in WSNs.

In **Chapter 3** I report on new cross-layer methods for WSN lifetime optimisation by integrating time synchronisation into the routing layer. Two algorithms are proposed: CLEAR – a Cross Layer Efficient Architecture for Routing and RISS – a Routing Integrated Synchronisation Service. The performance of these techniques in terms of energy optimisation and synchronisation precision is compared with the prevailing routing and synchronisation algorithms.

In **Chapter 4** I describe new techniques for optimising the energy efficiency of the routing protocol which incorporate information about the network topology into the network layer using a cross-layer design approach. A new method of intra-cluster topology selection, called SCALE – Smart Clustering Adapted LEACH, is presented which extends the network lifetime. This chapter also exploits the influence of the hop distance on the energy efficiency of a route and as a result of this investigation an algorithm to determine optimal location of the relay node is described. To conclude the chapter, a theoretical study of transition region is presented and a mechanism of avoiding routing over nodes located in this region is proposed.

In **Chapter 5** I conclude the thesis with a summary of this work along with some suggestions for future research.

CHAPTER 2

Routing in Wireless Sensor Networks

Routing is a necessary element of data networks, in particular of WSNs composed of many nodes where intermediate points have to relay packets from the source to the destination node. That is why routing is required, and can be defined for packet-switched networks as follows [47, 145]:

> Routing is the process of determining an end-to-end path between the sender and the receiver for a packet.

This process is supported by *routing protocols*, which in the traditional approach allow nodes, by exchanging information such as connectivity and link states, to build up a picture of the whole network so they can choose the best way to forward a packet. Based on this information *routing algorithms* determine the best path along which to forward a packet. The results of routing algorithms are used to create and update the *routing tables* that are used later in the *forwarding* process.

A different approach to routing is based on a so-called *data-centric* technique where instead of addressing individual nodes, data is the focus of attention. Data-centric networking [18, 25, 63, 70, 86] is of relevance in WSN applications as their interest may be not so much in the identity of a particular sensor node but rather in the actual information reported about the surrounding environment. Hence, in a data-centric routing protocol:

> "flow of information – from sender to receiver – is determined by the specific interests of the receiver rather than by an explicit address assigned by the sender. With this communication pattern, receivers subscribe to information that is of interest to them without regard to any specific source (unless that is one of the selection criteria), while senders simply publish information without address it to any specific destination" [19].

In both approaches to routing in WSNs, node-centric and data-centric, a key objective is to save energy. Nodes typically have very limited energy resources which are drained mainly by the transceiver components. In this chapter numerous challenges and approaches in designing routing paradigm for WSNs are presented. Because some of the techniques used for routing in WSNs are borrowed from other networks, the evolution of routing with many corresponding examples is also described.

Also discussed are three related topics: energy efficiency optimisation, the cross-layer design approach, and hop-distance optimisation.

2.1 Routing legacy in existing networks

There are many different views on how the routing in a WSN should be done [7, 8]. Authors design novel protocols but they also avail of solutions from other technologies. In this section I provide a short overview from a historical perspective of how network routing and especially routing in WSNs evolved.

2.1.1 Telecommunication networks

Circuit switching is the oldest technology which prevailed in communication networks for many decades. Originally it was designed for telecommunication networks to carry voice communication and consisted of establishing physical circuits from sources to destinations of phone calls. Each of these circuits was reserved and dedicated for the users at both ends for the whole duration of the call.

2.1.1.1 Routing in telephone networks

Initially, the telephone network relied on static, preconfigured and computed off-line routes. The patterns depended upon network topology and provisioning of traffic demand, but they remained independent of the state of the network or time of day. To minimise the impact of eventual route failure or network overcharge, the switches were provided with multiple routes to each destination. Manual intervention was necessary to reconfigure the set of paths if necessary.

The main advantage of this static routing was due to the complete control over the routes selected and reduced switch computation requirements. However, these systems adapted slowly to unpredicted events and changing network state. Also they required frequent human intervention and a substantial amount of switch memory to store the configured multiple routes per destination. Thus dynamic routing protocols were introduced to mitigate the inefficiency of former protocols.

Their expansion was hindered by the limited processing capacity of switches and also by the concern of telephone companies about the eventual economic implications due to the incorrect routing decisions made by the network itself. However the introduction of stored-program-control switches with high-speed processing capabilities, speeded up that progress. Finally the companies recognised the advantages of dynamic routing, which are reduced network trunking and also operating cost. It has been estimated that dynamic routing can increase the capacity of the telephone network by up to 30% (compared to static routing) [83].

A simple dynamic routing approach was applied in *Dynamic Alternative Routing (DAR)* [144]. DAR is an adaptive call-routing strategy that stochastically selects an alternative route when a direct route is not available and uses local information about the loading of outgoing trunks to determine the feasibility of selected routes. One of the main advantages of DAR is its speed of response due mainly to its distributed operation. To achieve processing simplicity, the protocol does not process a path for a call on the basis of the overall network traffic.

This type of strategy was implemented in *Network state dependent* protocols which are an important subset of dynamic routing strategies. They estimate traffic congestion and try to establish connections in a way which minimises the probability of future call blocking. *Dynamic Nonhierarchical Routing (DNHR)* is an example of such a protocol. It uses extensive off-line calculation to select, for a particular time of the day, a set of alternative routes for every pair of core switches. DNHR responds slowly to the traffic fluctuations because routing patterns are constructed for a period of time on the basis of historical information. *Dynamically Controlled Routing (DCR)* is a centralised protocol which selects routes based on analysis of network status reports sent periodically by core switches. However this protocol is very susceptible to failure of the control processor because of the need for frequent updating of routing tables [68].

The evolution of telephone networks led to the development of Asynchronous Transfer Mode (ATM) technology which uses different routing protocols.

2.1.1.2 Routing in ATM networks

Telephone networks were initially pure analog systems carrying voice data on mechanically interconnected wires. The quality and rate advantage of digital transmission motivated introduction of digital transmission on phone networks. Also in the mid 1980s the telephone industry observed an increased market demand for other services *e.g.* videoconferencing, internet access, data transfer *etc.* Telephone companies thus decided to build an integrated voice/data network that they called Integrated Services Digital Network (ISDN) [5]. Soon, the transfer rate defined in the standard turned out to be inadequate to assure the success of ISDN and thus CCITT issued an improved version called the B-ISDN where the B stands for "broadband".

By the end of the 1980s, Asynchronous Transfer Mode (ATM) [75] was developed as a promising technology to carry both synchronous voice and asynchronous data service and it was recommended for B-ISDN. ATM represents the transition from digital circuits to the packet based telecommunications networks. It combines benefits of circuit switching (*e.g.* constant transmission delay and guaranteed capacity) with those of packet switching (*e.g.* flexibility). Routing consists of establishing a so-called virtual channel between source and destination at the beginning of the connection and guiding packets (called cells) over it. There are two main routing protocols in ATM networks, Private Network-Node Interface (PNNI) and Interim Interswitch Signalling Protocol (IISP).

IISP provides a static routing solution and is based on manually configured routing tables. Thus for smaller systems it is very simple to deploy and also can be used to connect proprietary implementations of PNNI. However, for large networks it is prone to errors and time-consuming to configure.

In contrast to IISP, PNNI supports QoS and crankback. It provides two significant services: network topology discovery and call establishment. It is a hierarchical, dynamic link-state, source routing protocol. So upon receiving the call request a source router references the PNNI routing table to determine a path to the intended destination that is capable to support the QoS requirements specified by the caller. The connection message is then forwarded to the destination along the potential path and if sufficient resources are available on every intermediate node, the transmission can start. Otherwise the crankback occurs and a new path is computed. If it does not satisfy the request requirements, the connection is refused.

Numerous telecom companies have deployed ATM networks. However, it has failed to gain wide use as a LAN technology and its great complexity has hampered its full deployment as the single integrating network technology in the way that its inventors originally intended.

2.1.2 Data networks

Packet switching technology was designed in the 1960s as a means for providing cost-effective and efficient data communication between large computers and remote users. In a data (packet-switched) network packets are routed between nodes over links shared with other traffic. The evolution of this type of network has been driven primarily by advances in computer technology. Historically, three concurrent efforts contributed to the rapid development of packet-switching technology:

- ARPANET [113, 119, 127], a network created in 1969 by Advanced Research Projects Agency (ARPA) for the US Defence Department and initially interconnecting four computers. During the 1970s, the ARPANET grew, connecting research institutes and laboratories throughout USA and Europe. In 1990 the ARPANET was retired and its functions had already been replaced by various higher speed networks.

- Computer time-sharing companies, eager to expand their computing resources, developed their own packet-switched networks (*e.g.* TYMNET) [131] to provide remote user access to their geographically distributed machines. Most such networks migrated to IP technology and are now considered obsolete.

- Computer manufacturers designed integrated proprietary systems for data communication (*e.g.* IBM's SNA [99]) to add functionality to their computers and peripherals. Although many such networks are still in operation, they are also being replaced by IP technology.

The purpose of a network influenced the specific design of its routing strategy. The routing algorithms used in these networks all turn out to be variants of shortest path algorithms that route packets from source to destination over a path of least cost (*e.g.* minimum delay). Nowadays, in general most data networks employ some type of shortest-path strategy to generate and select routes according to the network and user state. Thus I describe two least cost algorithms (distance-vector and link-state), which are the basis for routing procedures in many networks (including Wireless Sensor Networks).

2.1.2.1 Distance-vector routing

The Distance Vector (DV) technique was developed first and initially it was employed in data networks with the ARPANET project in 1969 [101]. It is also known as the Bellman-Ford algorithm after its creators [14]. In a DV algorithm each router knows the identity of every other router in the network. Each router maintains a DV, that is, a list of ⟨destination,cost⟩ tuples, one tuple per destination. The cost is an additive function of a current estimate of link costs (*e.g.* hop count) on the shortest path to the destination. A router periodically shares with its neighbours a copy of its DV. A receiver of that packet, determines whether it is possible to reach any destination with a smaller cost by sending packets through the sender. It can do so by comparing its current route cost to the destination with the sum of the cost to reach the neighbour and its neighbour's cost to communicate with the same destination.

The DV strategy was implemented in many routing protocols. One of the most common examples is the Routing Information Protocol (RIP) [59]. It was initially deployed with Xerox Network Services by Xerox. RIP is most commonly used as a routing protocol on intranets. The main advantage of the DV protocol is its small overhead and simplicity. This permits distributed and asynchronous operation and requires only locally available costs to compute and select routes. However, the Bellman-Ford algorithm causes problems when the network architecture is unstable. It suffers then from a count-to-infinity problem [144] and does not prevent routing loops. Thus another class of protocols (called link-state) for data networks was developed.

2.1.2.2 Link-state routing

Link-state (LS) routing was developed for the ARPANET in 1978 by John McQuillan [100] as a solution to the instability of DV protocols in cases of changing network connectivity. Later this type of strategy was proposed for use in an ISO Intermediate System to Intermediate System (IS-IS) routing protocol [2]. The LS algorithm was also adapted by the Internet Engineering Task Force in the Open Shortest Path First (OSPF) protocol [107] for use in the Internet. Both protocols, IS-IS and OSPF, are broadly utilised for intra autonomous system[1] routing.

In a LS protocol a router periodically broadcasts its local view of the network, in terms of the properties of the links connecting it to neighbouring nodes. Then every router updates its dynamic map of the network which is essentially a database describing the network's components and their current interconnections. From this database routers compute routes for the traffic. This contrasts with the DV strategy in which every router shares its routing table with neighbours and this information serves to construct local routing paths.

The prime advantage of LS routing is that it reacts quickly to the network connectivity changes. It also requires a smaller overhead than the DV algorithm as in the earlier protocol nodes only broadcast information about links with their immediate neighbours whereas in the DV protocol routers share entire routing tables. The main disadvantage of LS is that it requires more computational and stor-

[1] An autonomous system is a collection of networks under the control of a single entity

age resources than the DV protocol. Both routing classes, DV and LS, are broadly used for routing in a single autonomous system. However for communication between autonomous systems so-called Inter-domain algorithms were developed.

2.1.2.3 Inter-domain routing

The need for distinct algorithms to route packets between heterogenous internets comprising multiple organisations emerged with the growth of the Internet. After being initially a research project it became in the late 1980s a worldwide communication infrastructure for diverse organisations with heterogenous goals and objectives. Thus, interconnecting these users requires extensive cooperation among disparate constituent networks.

Three protocols chiefly contributed to the evolution of inter-network routing, each based on the DV paradigm. The three protocols are the Exterior Gateway Protocol (EGP), the Border Gateway Protocol (BGP) and the Inter-Domain Routing Protocol (IDRP).

EGP was the first inter-domain routing protocol introduced in the Internet. It was designed in 1982 [129] and was deployed to deal with routing between ARPANET and sites attached to it. EGP introduced for the first time the concept of a domain, having a globally unique identifier. The protocol operations consist of three main tasks: neighbour acquisition, neighbour reachability and exchange of routing information. The former objective is achieved by a simple two-way handshake. Then to maintain the information about connectivity with neighbours, nodes periodically send HELLO messages and expect responses from adjacent routers. Finally, the routing information is shared between neighbours via a message including NLRI[2] data, the metric (whose definition is left to the designers of the autonomous system [129]), and network-layer address of an appropriate next-hop. EGP was sufficient for the initial needs of the Internet but when the network

[2]NLRI, network-layer reachability information, is a list of IP network numbers

architecture became more complex, its suitability deteriorated and it became obsolete. It was replaced in the late 1980s by BGP which addressed all the shortcomings of EGP.

The current version of **BGP** (v4) is the core routing protocol of the Internet. It allows an arbitrary interconnection of network topologies between autonomous systems in contrast to EGP which was intended to operate only within a strictly hierarchical inter-domain topology. Nodes exchange similar routing information as did EGP. Additionally, routers add to the packet the list of domains that the routing information has traversed so far. This mechanism prevents routing loops and is the reason for calling BGP a path vector protocol. Scalability is a key requirement for the inter-domain routing and therefore BGP provides a mechanism that allows reduction of the volume of the information that needs to be handled by routers. Routers assume that the network is hierarchical and therefore the NLRI of topologically close destinations can be aggregated by NLRI summarisation. BGP adopts a policy-based routing mechanism where each domain applies local rules to select a route and to decide whether to propagate this route to neighbouring domains. Although there has been a continuous improvement of BGP, it has some shortcomings. The number of domains is limited to 65536 which is insufficient for very large internets. Besides, BGP provides a single path to a destination and therefore it excludes the ability to support multiple routes with different performance characteristics.

IDRP was developed to accommodate the growth in the number of networks and users. It shares many features with BGP (path-vector routing, aggregation of NLRI, variable address lengths) and some of them are improved. A significant effort was undertaken to increase the scalability of IDRP in heterogenous, multiprotocol internets. As a result IDRP allows up to 2^{160} domains, in comparison to 2^{16} the maximum of domains allowed by BGP. IDRP was a major divergence from the BGP development track, the *de-facto* standard of the Internet, and was not adopted by industry. One primary reason for this was that IDRP was incompatible with the BGP development track, which was operational, mature, and commercially supported by major IP router vendors with running code in the Internet. In addition, the IETF[3] considered the complications of IDRP QoS routing enhancements less attractive relative to the simpler BGP paradigm.

2.1.3 Wireless multihop networks

The research on wireless packet networks was initiated in the 1970s with the first successful network based on packet radio developed at the University of Hawaii in 1971 called ALOHANET. This system enabled computer sites at seven campuses spread out over four islands to communicate with a central computer via radio transmission. Although it assumed a simple star topology of the network, many solutions and principles incorporated in its routing protocol are still in use today. Then DARPA invested significant resources during the 1970s and 1980s to develop networks using packet radios for military applications. Over time, packet radio networks found commercial applications in supporting wide area wireless data services (*e.g.* email, file transfer, web browsing) in the 1990s.

The development of data wireless networks led to the classification of those into Wireless Mesh Networks (WMNs) and Mobile Ad-hoc Network (MANET). They are designed for different applications and deployment objectives and thus use different routing protocols. However, both these types of networks share many properties with WSNs and have substantially influenced the development of WSNs routing protocols.

2.1.3.1 Mobile Ad-hoc Networks routing

MANET originated from the DARPA Packet Radio Network [77] developed in the 1970s. In 1983 DARPA established a successor to the packet-radio program which

[3] Internet Engineering Task Force is the body that develops and promotes Internet standards

it called Survivable Adaptive Networks (SURAN) [144]. MANET is formed by mobile devices that can dynamically move and reorganise themselves and communicate over wireless links. Because of the importance of routing protocols in dynamic multi-hop networks, a lot of mobile ad hoc network routing protocols have been proposed in the past.

The development of routing protocols for MANET was boosted by the formation of the IETF MANET working group and publication of its charter in 1997 with the objective of developing a solution framework for routing in an ad hoc network. We can distinguish three main groups of routing protocols for MANET:

- Proactive, table-driven routing protocols[4]:
  - Optimised Link State Routing Protocol (OLSR) [29]
  - Fisheye State Routing (FSR) [115]
  - Destination Sequence Distance Vector (DSDV) [118]
- Reactive, source initiated, on-demand routing protocols[5]:
  - Dynamic Source Routing (DSR) [76]
  - Ad hoc On-demand Distance Vector Routing (AODV) [117]
  - Temporally Ordered Routing Algorithm (TORA) [114]
- Hybrid routing protocols[6]:
  - Zone Routing Protocol (ZRP) [51]

[4]Proactive protocols are routing algorithms that will periodically and continuously update routes in the network so that when a packet needs to be transmitted, routes to the destination are already known and packets can be forwarded straight away.

[5]Reactive protocols compute a route only when it is needed.

[6]Hybrid routing protocols try to combine the advantages of both the philosophies: proactive is used for near nodes or often used paths, while reactive routing is used for more distant nodes or less often used paths.

Optimised Link State Routing Protocol (OLSR) is a very popular proactive, LS routing approach for wireless Ad hoc networks designed at INRIA and standardised at IETF [29]. The key concept of OLSR is to minimise the protocol overhead by reducing the number of retransmitted link state packets. Each node selects a subset of its neighbours (called multipoint relays (MPR)) in such a way that all 2-hop neighbours receive broadcast messages even if only MPRs nodes retransmit the packets. The MPR set is computed by every node independently, solely based on the received local topology. Additionally, the OLSR reduces the overhead because nodes need to share only the link state information to all MPR selectors for the computation of shortest paths. Every node periodically broadcasts its link state information by topology control messages. The frequency of topology control messages increases when topology change in the network has been detected. The OLSR routing table that contains entries for all reachable destinations in the mesh network is computed with a classical shortest path algorithm (*e.g.* Dijkstra algorithm). The initial OLSR protocol does not take into account the link quality for the route selection algorithm. This assumes that links are bimodal (either working when packets from a given node can be heard or failed if not), which is not necessarily the case in wireless networks and may result in a high packet loss. Later implementations of OLSR (*e.g.* Radio Aware-OLSR (section 2.1.3.2)) solve that problem by using radio-aware metrics in forwarding path and MPR set calculations. Another disadvantage of OLSR is that OLSR floods the link state database unreliably and may cause transient loops if the link state database becomes inconsistent because of the packet loss.

Fisheye State Routing (FSR) is a proactive protocol, based on the LS paradigm with significantly reduced overhead caused by the update of network topology information. The idea behind the protocol is based on the fish eye lens which catches the pixels near the focus with high detail, and the detail decreases with distance from the focal point. Similarly, FSR maintains accurate distance and path quality information about immediately neighbouring nodes, and progressively reduces detail as the distance increases. This is achieved by varying the periodicity of the LS messages with the scope of the destination. The nodes closer to the source of LS packet receive topology information more frequently than faraway nodes. This is done by incrementing TTL[7] of messages for each flood until the maximum value before it continues with the initial, small value. FSR exhibits very good scalability because it does not strive to keep all nodes in the network on the same knowledge level about link states. Although FSR limits the topology information shared among nodes, the packets are routed correctly because the route information becomes more and more accurate as the packet gets closer to the destination. The main disadvantages of FSR are its large processing overhead and routing table storage complexity.

Destination Sequence Distance Vector (DSDV), a table-driven routing protocol based on the classical Bellman-Ford algorithm, was developed in 1994. The main contribution of the algorithm is to prevent the routing loop problem. Every node maintains a routing table in which all possible destinations within the network and the number of hops to each destination are recorded. Each entry is marked with a sequence number assigned by the destination node. Sequence numbers are used to distinguish stale routes from fresh ones and to avoid the formation of route loops. Routing tables are constructed on the base of the routing updates broadcasted periodically by every node. To reduce protocol overhead, these routing updates can employ two possible types of packets. The first is known as a full dump. This type of packet carries all available routing information and can require multiple **NPDUs!**s (**NPDUs!**s). During periods of occasional movement, these packets are transmitted infrequently. Smaller incremental packets are used to relay only that information which has changed since the last full dump. DSDV is not suitable for highly dynamic networks because whenever the topology of

[7] Time-To-Live is a value in a routing header which tells the network router whether or not the packet has been in the network too long and should be discarded.

network changes, a new sequence number is necessary before the network re-converges, which can take a significant period of time.

Dynamic Source Routing (DSR), one of the pioneering routing solutions for MANET, is a reactive protocol that utilises a source routing algorithm. DSR consists of two major phases: route discovery and route maintenance. When a source node wants to send a packet, it firstly consults its routing cache. If the required route is available, the node inserts it into the header of the packet. Otherwise, it broadcasts the route discovery packet. Receiving that packet, a node checks its route cache. If the node does not have routing information for the requested destination, it forwards the packet with its own address appended to the route record field of the header. When the request reaches the destination or an intermediate node has routing information to the destination, a route reply packet is sent. It comprises addresses of nodes that have been traversed by the request packet eventually concatenated with the route from the intermediate node's cache. A *route error* packet is generated when a node discovers link failure. Then all the nodes remove from the cache all routes containing the broken link. DSR, as a reactive protocol, eliminates the overhead due to the periodical flooding of the network with route updates. This approach however increases the connection setup delay and its performance degrades rapidly in a mobile environment. Also DSR requires considerable routing overhead because every data packet contains complete routing information.

Ad hoc On-demand Distance Vector routing (AODV) is a popular, reactive routing protocol for MANET which has been standardised by IETF. AODV uses many solutions implemented in DSDV. The main difference between these protocols relies in the route calculation task which is performed only when necessary in case of AODV. When a source node wants to send a message to some destination node and does not already have a valid route to that destination, it broadcasts a route request (RREQ) packet to its neighbours, which then forward the request to their neighbours, and so on, until either the destination or an intermediate node with a route to the destination is reached. Then a route reply (RREP) packet is created and forwarded back to the source. The RREP follows the reverse path of the respective RREQ as AODV only supports the use of symmetric links. Upon receiving the RREP packet, every intermediate node updates its next-hop routing table entry for the corresponding destination node. When a node discovers a link failure, it broadcasts a route error (RERR) message to its neighbours, which then forward the message to nodes whose routing tables may be affected by this change. Then, the route discovery procedure may be re-initiated if the route is needed. The disadvantage of AODV is that multiple RREP packets generated in response to a single RREQ packet can cause a heavy control overhead.

Temporally Ordered Routing Algorithm (TORA) is a reactive, highly-adaptive, distributed routing algorithm based on the concept of link reversal. The key idea of TORA is to limit the propagation of routing control messages to a very small set of nodes near the occurrence of a topological change. The operation of the protocol is composed of three main functions: route creation, maintenance and erasure. During the route creation and maintenance phases, TORA assigns to every node a "height" metric in order to construct a Directed Acyclic Graph (DAG) rooted at the destination. The destination of the message has a value of height 0 and values assigned for remaining nodes are proportional to the distance from the destination. Then, like water flowing, a packet goes from upstream to downstream according to the height difference between nodes. In the event of a link failure, when a node loses its last downstream link, it generates a new reference level which results in the propagation of that reference level by neighbouring nodes and in consequence the reversion of links directions reflecting the new reference level. The erasing operation of outdated entries from routing tables in TORA consists of flooding CLR packets and dropping invalid routes. TORA may suffer from oscillations especially in situations when multiple nodes detect link failures. However those oscillations are temporary and are followed by a route convergence. TORA requires clock synchronisation of nodes which may be an energy demanding operation especially in large networks.

Zone Routing Protocol (ZRP) is a hybrid routing protocol that implements an algorithm which divides the network into overlapping zones and then uses a proactive routing within the zones and reactive routing between them. Zones are selected according to the hop distance between mobile nodes. Most existing proactive routing approaches can be used for intra-zone routing. For inter-zone routing the Inter-zone Routing Protocol (IERP) is used which is very similar to DSR. The hybrid routing approach decreases the route setup delay of reactive routing schemes and reduces the control overhead of proactive routing protocols. ZRP causes large overlapping of routing zones and thus it generally suffers from bigger overhead than other hybrid protocols like Hybrid Ad hoc Routing Protocol (HARP) [111] or Zone-based Hierarchical Link State routing protocol (ZHLS) [74].

2.1.3.2 Wireless Mesh Networks

The main factor driving the development of Wireless Mesh Networks (WMNs) was the expansion of the Internet in the 1990s. WMNs were intended to extend a connection to Internet for wireless devices. The network is formed of static wireless relay nodes providing a distributed infrastructure for mobile client nodes over a partial mesh topology. The main difference between a MANET and a WMN lies in the mobility of nodes and network topology. MANETs are characterised by the lack of static infrastructure and a highly dynamic topology whereas WMNs use multihop wireless relaying over a relatively static partial mesh topology for its communication. In 2004 the 802.11s Task Group (responsible for standardisation of mesh networking in Wireless Local Area Network (WLAN)) was created to address the issue of lack of interoperability between equipment from different vendors and also to further the ability and ubiquity of mesh networking.

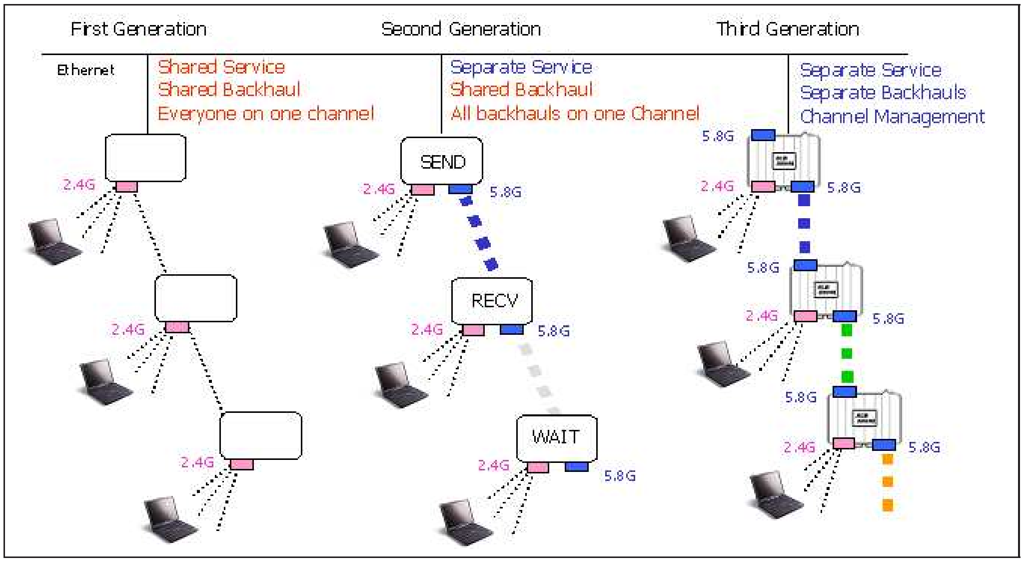


Figure 2.1: *Three generations of Wireless Mesh Networks*

Over the years WMNs have seen three generations of technology (see Figure 2.1) of which the second and the third were defined by the 802.11s Task Group (the first generation corresponds to the early WMNs systems). Each of them incorporated iterative improvements and allowed for greater scalability and higher network throughput and reduced latency. These generations are described below:

- In the first generation the radio mesh uses one radio channel both to service clients and to provide the mesh backhaul. This architecture is very inefficient since many users compete for the bandwidth and mesh nodes have to listen frequently to the channel prior to every transmission.

- The second generation of WMNs uses separate radio channels for servicing the clients and for mesh backhaul. Most currently available products use this architecture and it considerably improves network performance in comparison to the first generation WMNs.

- The third generation of WMNs, also uses different channels for service and backhaul functionalities, but now these channels are managed dynamically so that there is no interference between occupied links. This solution provides improved performance as it preserves bandwidth and reduces latency over multiple mesh hops.

Some MANET routing protocols are widely used in WMNs due to the similarities between these architectures. However, many routing protocols have been designed particularly for WMNs based on the unique properties of WMNs (mulithop, wireless, mesh architecture with fixed nodes, reliable with high throughput links, self-healing and self-configurable network).

Existing WMNs routing protocols can be classified as being position-based or topology-based. Furthermore, the topology-based algorithms are divided into three categories: proactive, reactive and hybrid (see section 2.1.3.1). This classification with corresponding examples is shown below:

- location-based protocols, *e.g.* Greedy Perimeter Stateless Routing (GPSR) [81]
- topology-based protocols
  - Proactive, table-driven routing protocols, *e.g.* Radio Aware Optimised Link State Routing (RA-OLSR) (based on [29, 115])
  - Reactive, source initiated, on-demand routing protocols, *e.g.* Multiradio Link Quality Source Routing (MRLQSR) [42]
  - Hybrid routing protocols, *e.g.* Hybrid wireless mesh protocol (HWMP) (based on [117])

Greedy Perimeter Stateless Routing (GPSR) is one of the first practical position-based routing protocols for wireless networks. It exploits the correspondence between geographic position and connectivity in a wireless network, by using the positions of nodes to make packet forwarding decisions. GPSR combines greedy forwarding[8] (to forward packets to nodes that are always progressively closer to

[8]In a greedy forwarding algorithm packet is sent to the neighbour closest to the destination.

the destination) with face routing[9] as a fallback (after bypassing the disconnected region the greedy forwarding resumes). The main advantage of GPSR is that forwarding decisions are made using local information and there is no need to maintain routing tables for all nodes. However, GPSR requires that nodes know their locations which may be difficult to obtain, especially in large networks.

Radio Aware Optimised Link State Routing (RA-OLSR) is a proactive, link-state and optional routing protocol of the IEEE 802.11s standard. It is an adaptation of the OLSR protocol (section 2.1.3.1) to the WMN environment incorporating some features of the FSR paradigm (section 2.1.3.1). Packets are forwarded along the shortest path according to an arbitrary, radio-aware metric (*e.g.* an air-time metric[10]) instead of using a hop-count metric as in OLSR. To propagate metric link cost information between mesh nodes, a link metric field is associated to each reported neighbour in hello messages and topology control messages. To reduce the related control overhead, RA-OLSR uses the concept of MPRs (similarly to OLSR) and optionally controls and reduces the message exchange frequencies based on Fisheye scopes (see section 2.1.3.1).

Each mesh node maintains a local association base (LAB) that contains all legacy client stations associated with this mesh node. It broadcasts local association base advertisement (LABA) messages periodically, in order to distribute the association information in the mesh network. Each receiver of LABA messages stores this information in the global association base (GAB). The content of both LAB and GAB is used in the construction of the routing table and provides routes to legacy client stations associated with mesh nodes. To reduce overhead, it is possible to advertise only the checksum of the blocks of the LAB. If there is a mismatch between a received checksum and the checksum in the GAB, the node

[9]In face routing the network graph is logically segmented into so-called faces, where the considered links do not cross each other. Packets can proceed out of a local minimum by being forwarded around these faces toward the destination.

[10]An air-time metric reflects the amount of channel resources consumed for transmitting a frame over a particular link.

requests an update of the corresponding block of the LAB of the originating node. RA-OLSR inherits the stability of a link-state routing protocol and has the advantage of having routes immediately available when needed due to its proactive nature.

Multiradio Link Quality Source Routing (MRLQSR) is a reactive protocol based on the DSR algorithm (section 2.1.3.1) and developed by Microsoft for static community third generation WMNs. The main contribution of MRLQSR is the use of a new routing metric called weighted cumulative expected transmission time (WCETT) which takes into account link quality, channel diversity, and minimum hopcount. It can achieve a good trade-off between throughput and delay because it considers channels with good quality and channel diversity at the same time. Neighbour discovery, link weight information propagation and path finding processes are similar to the DSR protocol. MRLQSR identifies all nodes in the WMN and assigns weights (according to the WCETT metric) to all possible links.

The main advantage of MRLQSR is the improved throughput performance compared with the throughput achieved by other multiradio routing metrics (*e.g.* it was found that WCETT outperforms the ETX[11] routing metric by about 80% [159]). This throughput advantage is because MRLQSR considers a trade-off between end-to-end delay and the path throughput for candidate paths. One of main disadvantages is that the use of multiple radios on a single node may consume substantial power and hence the routing metric should effectively look into energy-efficient routes when used in mobile WMNs.

Hybrid wireless mesh protocol (HWMP) is the default routing protocol of IEEE 802.11s compliant networks. The foundation of HWMP is an adaptation of the reactive routing protocol AODV (section 2.1.3.1), called radio metric AODV (RM-AODV), to layer 2 and to radio-aware metrics. HWMP combines ad hoc and span-

[11]The expected transmission count (ETX) metric assigns a weight to each link that corresponds to the expected number of transmissions required to successfully transmit a packet over the link.

ning tree-based routing features that incorporate proactive and reactive methods in the following way:

- The former method is applied if a node in the WMN is optionally configured as a root node and then other nodes proactively maintain routes to the root using topology discovery primitives and distance vector methodology identical to the RM-AODV protocol. When a node wants to send a packet, and has no route to the destination, it may send a frame to the root. Then it looks up the routing and bridging tables to see if the packet is intended for a node within the mesh or outside. It forwards the message appropriately back to the mesh or its uplink. If it finds the entry inside the mesh, it sends the frame to the destination parent mesh node which in turn will initiate an on-demand, optimal route discovery between the destination-source pair for all subsequent frames sent between them.

- The latter approach uses an RREQ and a route reply RREP mechanism (borrowed from AODV (see section 2.1.3.1)) to establish routes between two mesh nodes. The main difference from AODV is that HWMP works at layer-2 and uses MAC addresses. Apart from this adaptation, it uses the following mechanisms of the original AODV protocol: route discovery, route maintenance, best candidate route caching, sequencing, route acknowledgment, and route errors.

The main advantage of HWMP is the scalability, due to its hybrid nature, it combines the flexibility of on-demand route discovery with the option of efficient proactive routing to a mesh node. Another benefit of HWMP is its reduced protocol overhead. The protocol reduces flooding by sending a single RREQ for multiple destinations simultaneously. It also enables proactive maintenance of routes to popular destinations (*e.g.* the root node) and to do that each active source node sends a periodic RREQ message (maintenance RREQ) for those destination(s).

The routing protocols for wireless multihop networks significantly influenced the development of the network layer in WSNs. In the early 1990s many researchers proposed to deploy on WSNs routing protocols designed for routing in ad-hoc networks. However the exponential growth of research in WSN in subsequent years [7, 69, 80, 130] showed the inefficiency and weaknesses of this approach due to differences between these wireless network types. Hence many routing algorithms have been designed specifically for use in WSNs.

2.2 Routing in Wireless Sensor Networks

WSNs were developed to monitor and interact with the surrounding environment. Although a WSN is composed of simple nodes with limited performance the network is able to execute complex tasks due to the interaction between sensor nodes. To exchange data, sensor nodes need to forward packets through their neighbours because their energy resources and transmission range are limited. Hence, routing is required in most WSNs. In early stage of WSN development many researchers proposed using packet routing protocols designed for other networks (*e.g.* MANET). Over time many routing solutions tailored for WSNs were designed. This section describes the challenges faced in developing novel routing techniques and solutions for routing in WSNs.

2.2.1 Properties of WSNs influencing routing

WSNs, since they are packet-switched wireless systems, share many properties with the other networks, and are especially similar to MANET in many aspects (*e.g.* self-organisation, multihopping, distributed systems and lack of fixed infrastructure). These similarities and other characteristics of WSNs influencing design of routing paradigm are specified below:

- A WSN is generally composed of a large number of nodes which are randomly and densely deployed in a remote environment. Thus, it is impossible to build a global addressing scheme for a WSN because it would be inefficient and difficult to manage.

- Also, as a result of the manner of deployment and the usually limited potential for maintenance afterwards, WSNs are self-configurable, self-healing and operate in a distributed manner.
- The nodes have severely limited energy, processing, communication, and storage resources. Also, in contrast to MANETs, sensor node batteries are not recharged once deployed.
- The mobility of nodes is limited. Nevertheless the topology and connectivity of the network is dynamic because of limited node resources and their constant exposure to a harsh environment which may result in events such as the malfunctioning of a node, exhaustion of its batteries or its transceiver being asleep.
- WSNs are mainly designed to collect information with data streams in one direction, usually from multiple regions (sensors) to a particular base station. This property distinguishes WSNs from other networks (especially MANET) where data flows are more irregular.
- The generated data traffic has significant spatial correlation and redundancy since multiple adjacent sensors may produce similar output.

These characteristics are at the source of numerous challenges when designing a routing protocol for WSNs.

2.2.2 Challenges in WSN routing

The hardware advances which allowed the development of WSNs did not solve all the problems related to the design of robust protocols for these systems. Previously mentioned properties (see ref. 2.2.1) and other constraints of WSNs make the development of a routing paradigm for those networks very difficult with currently available IC[12] components. Below we detail the challenges which need to be faced when developing such protocols.

The expected lifetime of a WSN needs to be between 1 and 10 years for a typical application [146]. The amount of energy stored depends on the battery size and because a WSN is composed of very small nodes, their energy resources are very limited. This imposes tight constraints on the operation of sensor nodes. The transceiver is the element which drains most power from the node [46]. Thus the routing protocol will significantly influence the lifetime of the overall network. Also, a WSN node has a dual role: it acts as a source of sensor samples and as a relay. The death of some nodes may cause significant topological changes and may require reorganisation of the network. To minimise energy consumption, routing algorithms proposed in the literature for WSNs employ some well known energy efficient routing tactics as well as approaches specific to WSNs such as clustering, different node role assignment, data aggregation and in-network processing and data-centric methods.

Scalability is an important issue in WSNs [44]. The routing protocol has to be efficient in extremely large networks composed of even thousands nodes. The difficulty of this task is exacerbated by the nodes with very limited storage and processing resources.

The computational and storage capabilities of nodes significantly limit the feasible functionality of the routing protocol [35]. Hence, lightweight and simple versions of routing algorithms must be developed for WSNs. The problem of limited network capability can be solved with a little cost when only some of the sensor nodes are equipped with larger memory or faster CPU. This network heterogeneity has to be considered when designing a routing protocol.

[12]IC-integrated circuit

Node heterogeneity has to be taken into account when developing a routing protocol for WSNs [89]. There can be two main reasons for having a network with nodes of different capabilities. It is possible to increase performance of the network for reduced cost by deploying more powerful nodes which may have a role of uniformly distributed cluster-heads or may be relays situated close to the base station. Also differences between sensor nodes may arise over time due to the operation of the network. Some sensor nodes may execute more resource-intensive tasks which results in faster discharge of their batteries. Hence, the routing protocol should avoid relaying packets through sensor nodes with limited battery lifetime in order to compensate for energy heterogeneity among sensor nodes.

Node deployment in WSNs is application-dependent and can be either randomised or manual (deterministic). In the former case, nodes are scattered randomly and routes must be determined in a distributed manner [1]. In the latter case, the sensors are manually deployed and packets can be forwarded through predetermined paths [36]. However, even in this case routing should be decentralised when the network becomes large.

Fault tolerance must also be considered for packet routing [79]. Due to the exposure of nodes to harsh environments, they can fail unexpectedly. However, the failure of a sensor node should not affect the overall operation of the network. The routing algorithm should construct redundant paths or be capable of constructing a secondary route in a timely fashion in cases of link failure.

The communication range significantly influences network operation [22, 40, 67, 116, 156]. Nodes can vary their RF output power and in consequence the energy drained from their batteries during transmission. Sending packets with reduced power, over shorter distances, extends the node lifetime but also increases packet latency because there are more relays on the route to the base station[43, 66]. Conversely, when transmission hops are enlarged, the overall energy used for processing packets by intermediate nodes is reduced but interference may occur due to the broader range of RF signal [54].

Quality of Service (QoS) characterises a set of service requirements to be met when transporting a packet stream from the source to its destination in traditional networks [31]. However, the QoS constraints required by the application of WSNs may be very different and traditional end-to-end QoS parameters may be insufficient to describe them [21, 148]. In addition to traditional attributes such as data latency we can imagine others like: event classification error (if events are not only to be detected but also to be classified, the probability of error in classification must be low), event detection delay (the delay between detecting and reporting an event to the base station), tracking accuracy (in the case of tracking applications the reported position of an object should be as close to the real position as possible), missing reports (the probability of undelivered reports by an application that requires periodic reporting).

Mobility problem can be encountered in some WSNs applications [79]. Nodes can be fixed to moving objects in which case packet routing becomes more complicated. Also in some situations base stations can be mobile, which also needs to be taken into account when designing the routing paradigm.

2.2.3 Design approaches

Many techniques used for routing in WSNs are known from conventional communication networks but also many of them are fresh and specific to WSNs. Below is the list of different routing concepts with corresponding examples which are detailed in the following section.

2.2.3.1 flooding vs. gossiping vs. route estimation

When the transmission of a packet from sensor to the base station requires multi-hopping, the simplest way to reach the base station is then to flood [58] the network, *i.e.* send an incoming packet to all neighbours. To limit endless retransmissions, a node should forward only the packets that it has not yet seen. Flooding guaranties delivery of the packet to the base station if the source and the base station are not in disconnected regions. However, it has many drawbacks, such as implosion (caused by duplicated messages sent to the same node) and overlap (occurring when multiple nodes sensing the same region send similar packets to a common neighbour). It is also resource intensive. The overhead can be reduced by forwarding each packet to an arbitrary neighbour instead of to all of them. This type of communication is called gossiping [58]. It considerably increases the packet latency and does not guarantee its delivery to the base station. Both of these forwarding techniques are simple but their performance in terms of delay or throughput is very limited. This is mainly due to the lack of route prediction and also to the excessive number of packet transmissions [7]. Hence, it is preferable to use techniques which determine the appropriateness of forwarding a packet to a neighbour. This can be represented numerically by the cost of sending a packet to the base station via a particular neighbour. This cost function can be expressed with different metrics *e.g.* the number of hops or the energy required to reach the destination. Each node stores these costs in a so-called routing table. Then it forwards packets through the neighbour which corresponds to the minimum cost.

2.2.3.2 direct communication vs. flat routing vs. hierarchical routing

We can classify routing protocols according to the network structure into three categories: direct communication, flat routing and hierarchical routing. In direct communication algorithms, nodes send packets directly to the base station. In large WSNs these protocols are not applicable because they are very energy inefficient and can increase channel access delay. Furthermore, in this scenario the number of packet collisions limits the capacity of these protocols.

In flat routing protocols [18, 25, 63, 70, 86] every node is treated equally. With these algorithms, when a node wants to send data it transmits a packet to one of its neighbours that is closer to the base station. In this type of routing, a packet's delay is proportional to the distance between the packet's source and the base station. Also, nodes which are close to the base station may die earlier than distant sensor nodes because they relay packets from these sensor nodes besides sending their own data.

Hierarchical routing protocols [61, 90, 94, 95] assign different roles to nodes. Sensor nodes are grouped into clusters with a cluster-head that has the responsibility of forwarding packets from the cluster towards the base station. Furthermore, cluster-heads may be assembled into higher layer clusters and so on. Packets travel from a lower clustered layer to a higher one until they reach base station. Such inter-layer data forwarding reduces packet delay because hop distance increases with the layer level and so it reduces the number of hops needed to reach the base station. Also, nodes can be assigned roles according to their computational and power resources. Consequently, more powerful nodes become cluster-heads and can perform additional tasks such as data forwarding and aggregation, and cluster management. The other nodes are only responsible for sensing in the proximity of the target and transmitting their own data. Hierarchical routing is much more scalable and efficient than the previous two routing models.

2.2.3.3 reactive routing vs. proactive routing vs. hybrid routing

Following the taxonomy of ad-hoc routing protocols, we can also classify routing algorithms for WSNs as being reactive [91, 133], proactive [20, 86] or hybrid [18, 70] (see section 2.1.3.1). In contrast to ad-hoc networks, energy and storage resources are very limited in WSNs. Hence, nodes belonging to large WSNs cannot afford storage space for large routing tables. Also, because proactive protocols require significant energy to update routes, reactive and hybrid algorithms are preferred for WSNs.

2.2.3.4 broadcasting vs. multicasting vs. unicasting

WSNs routing protocols can be classified into three groups: broadcasting, multicasting or unicasting, according to the number of a packet intended receivers. Messages can be sent to uniquely identified nodes (unicasting), to all nodes in a network (broadcasting) or to a specified group of nodes (multicasting). Broadcasting and multicasting may be performed without significant additional cost in comparison to unicasting since a single transmission can spread a packet to multiple neighbours [135, 150]. However, reception of data drains power from the node and thus listening of the channel for a long time is energy inefficient.

2.2.3.5 self learning algorithms vs. static algorithms

Similarly to wired networks, we can distinguish two categories of routing protocols in WSNs: self learning algorithms (also called adaptive) [112, 158] and pre-setup (called static) algorithms [36]. The latter type of routing is easier to design and control and requires less resources in small networks with static nodes. However it is not suitable for WSNs, especially those of large size. As WSN nodes are prone to faults, the initial topology of the network may change during the operating time and the original routing configuration may become obsolete [158]. Also very often nodes have a little degree of mobility (because they are attached to a mobile platform) and static protocols cannot adapt to a new architecture. Finally, in WSNs with many nodes the calculation of all routes and their deployment on sensor nodes is impossible due to the complexity of such operations and storage limitations. Thus it is important for the routing protocol to be able to self-organise and reconfigure. To maximise network lifetime, the self-organisation phases must be short and energy efficient.

2.2.3.6 data-centric vs. node-centric protocols

Routing in conventional networks is based on a so called node-centric (or address-centric) approach where certain nodes are addressed by source nodes and packets should be delivered to these nodes. An alternative view of routing is that of data-centric networks [18, 25, 44, 63, 70] where the base station sends queries to nodes which are only implicitly described by providing certain attributes that these nodes have to fulfill (*e.g.* the base station can request a response from nodes that have observed a particular event). Data-centric routing allows very different networking architectures compared to traditional, address-centric networks. Firstly, it facilitates the implementation of data fusion and aggregation. Also, it enables simple representation of communication relationships – it does not require distinctions to be made between many-to-many, many-to-one, one-to-many, or one-to-one relationships as the set of participating nodes is only implicitly defined. Finally, it can improve performance and especially energy efficiency of a WSN because data-centric protocol can be implemented using purely local information about direct neighbours. A variety of approaches implementing data-centric networking have been proposed. In a publish/subscribe paradigm any node interested in a particular type of data can subscribe to it, and any node can publish data. Another possibility is to consider the WSN as a database from which the base station can retrieve information by formulating queries with an SQL-based language.

2.2.3.7 location aware vs. location-less

In location-based routing protocols [128, 152, 157] nodes are referred by means of their locations. This requires that nodes know their location, which can be learned at the expense of an additional energy cost. However, in most WSNs applications sensor nodes need to know their coordinates and hence routing can exploit this already available information. With location based routing algorithms, a substantial energy saving can be obtained by duty cycling sensor nodes in a localised manner such that all regions of the network remain connected to the base station via awake nodes.

2.2.3.8 time-driven vs. event-driven vs. query driven data delivery

The routing protocol energy efficiency and route calculation is influenced by the data reporting method which is application-dependent. We can distinguish three types of data reporting in WSNs: time-driven, event-driven and query-driven. In a time-driven approach [61, 63, 91] nodes report sensor samples according to a schedule, an approach particularly suitable for applications that require periodical environment monitoring. In the event-driven method [94, 95] nodes generate a packet as a result of changes in the value of a sensed parameter which is a consequence of an event occurrence. In a query-driven scheme [18, 70] the base station generates queries with a description of data that it is looking for. When the query reaches a node that possesses samples that matches this characteristic, it responds to the base station.

2.2.3.9 minimise energy per packet (or per bit) vs. maximise network lifetime

Achieving energy efficiency is one of the main challenges in designing a WSN protocols. The energy efficiency can be assessed in various ways. We can distinguish two dominant approaches to constructing an energy aware routing algorithm: minimising energy dissipation per packet or maximising network lifetime [79]. The former method involves minimising the total energy required to transport a packet over a multihop path from source to destination (including all overheads). Reducing the hop count will not necessarily achieve this goal as routes with few hops might include hops with large transmission power to cover large distances. This approach can lead to widely differing energy consumption on different nodes as some of them (especially those close to the base station) may be responsible for forwarding a significant proportion of packets. The latter method instead seeks to extend the network lifetime. The system lifetime can be defined in at least three ways: it can be the time until the first node dies or the time until network partition (when there are two nodes that can no longer communicate with each other) or the time until there is a spot that is not covered by the network.

2.2.3.10 long-hop vs. short-hop routing

The transceiver power in a WSN sensor node can be adjusted and thus the hop-distance and energy consumption can be modified [22, 40, 67, 116, 156]. As transmitted signal attenuation is proportional to a power function of the distance, it is generally more energy efficient to send packets over a route with many, short hops. However, short hops augment the number of relays and so the energy used for packet reception over a path increases. Also the packet delay and the energy required for packet processing on a route rises in this case.

2.2.3.11 intra-network data processing vs. pure gathering of data

WSN nodes report sensor samples to the base station. As the principal source of energy drain from the batteries is the transceiver, it can be very energy efficient to switch it off for as long as possible. This can be achieved by reducing the amount of data to be sent. There are many possible ways of doing that: data aggregation; distributed source coding and distributed compression; distributed and collaborative signal processing [18, 25, 61, 63, 70, 86, 94, 95, 140]. The former method consists of aggregating data from different sources at intermediate nodes on the route to the base station. This can be achieved by using functions such as average, min, max, median, *etc.* The challenges in this method include determining the impact of lost packets, where to aggregate the data and how long to wait for such results. In contrast to data aggregation which sacrifices information about measured values, the latter solution reduces the number of transmitted bits but without any information loss. It is achieved by exploiting the correlation between sensor samples from neighbour sensors or between consecutive readings of the same sensor. Distributed and collaborative signal processing consists of performing complex computations on a certain amount of data on the sensor node instead of at the base station if in return the number of bits to be transmitted can be reduced. An example of this concept is the distributed computation of a Fast Fourier Transform.

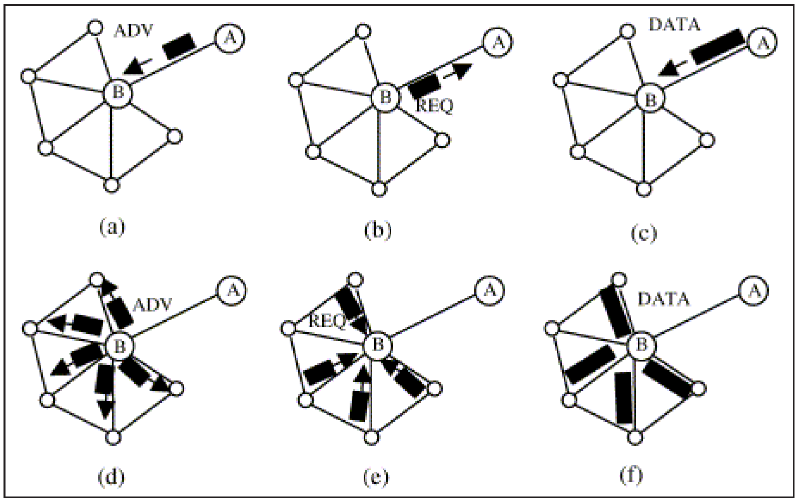
2.2.4 Prevailing routing protocols

In recent years many routing protocols for WSNs have been designed using various approaches. The most significant of these are described below for these various approaches.

2.2.4.1 Data-centric protocols

Sensor Protocol for Information via Negotiation (SPIN) is a family of data-centric dissemination protocols for WSNs proposed by Heinzelman *et al.* [63]. SPIN was designed for scenarios where each node disseminates its information to every node in the network assuming that they are potential base stations. Moreover, the amount of data at each node is large and thus SPIN-1 (the first of the SPIN family of protocols) assigns a high-level name to completely describe collected data, called meta-data, which is of small size relative to the data itself. This meta-data is used to negotiate the packet forwarding among nodes which is performed in three steps (see Figure 2.2).

First, a node that has obtained new data - either from a local sensor or from some other node - broadcasts an ADV message which includes the meta-data of this data. Its receiver can compare the meta-data included in the packet with its local knowledge and if the advertised data is yet unknown, it can send a REQ message to request the actual data. Otherwise, if the advertisement describes already received data, the ADV packet is ignored. Once the REQ message is re-



Source: A survey on routing protocols for WSN [7]

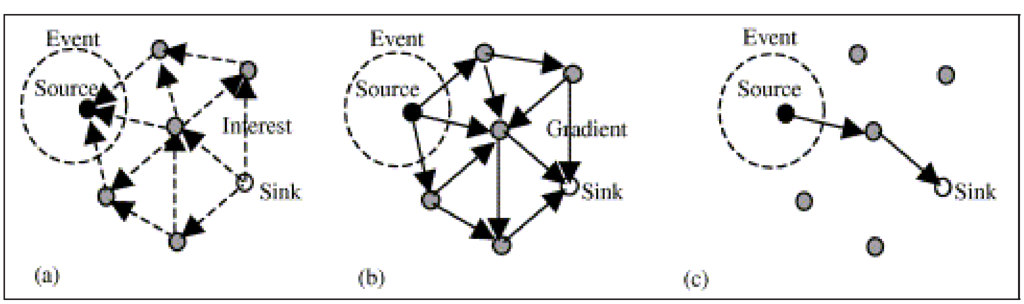
Figure 2.2: *The basic operation of the SPIN protocol. Node A starts by advertising its data to node B (a). Node B responds by sending a request to node A (b). After receiving the requested data (c), node B then sends out advertisements to its neighbours (d), who in turn send requests back to B (e,f).*

ceived, the sensor responds with actual data. The savings obtained with SPIN rest on the small size of the meta-data compared to the data itself and also on the elimination of redundant packets in comparison to conventional routing protocols based on flooding or gossiping. An extension to SPIN-1 is SPIN-2, which incorporates a threshold-based resource awareness mechanism in addition to negotiation. SPIN-2 checks the current energy level of a node and if it approaches a low threshold, it reduces the node's participation in the protocol (*i.e.* the node participates only if it can complete all the other stages of the protocol without going below the low energy threshold).

Kulik *et al.* proposed four additional protocols for the SPIN family: SPIN-BC, SPIN-PP, SPIN-EC and SPIN-RL [86]. SPIN-BC is designed for broadcast channels, that means all messages are sent to the broadcast address. This is more efficient because all neighbours can process the same packet while the disadvantage is that the nodes have to desist from transmitting if the channel is already in use. Also, SPIN-BC introduces random delay between reception of an ADV message and sending out a REQ packet. The node which hears the same REQ when its own REQ is waiting for transmission, does not send this packet. This mechanism eliminates redundant REQ copies of the request being sent again. SPIN-PP is defined for point-to-point communication. It assumes that all packets are successfully received and energy is not a constraint. The protocol works on a hop-by-hop basis with all unicast links. SPIN-EC is based on SPIN-PP and adds energy awareness. SPIN-RL was designed for lossy broadcast channels by incorporating two adjustments. First, each node keeps records of the ADV messages it receives and re-requests data if it has not arrived from the requested node within a specified time interval. Second, nodes limit the frequency with which they will resend data. Every node delays for a predetermined time period servicing requests for the piece of data which has been already requested.

SPIN protocols are claimed to be able to transmit 60 and 80% more data for a given amount of energy than conventional flooding or gossiping. However, SPIN is not scalable. Besides, nodes near the base stations are likely to die first due to their processing of messages from other nodes.

A different approach to data-centric routing is proposed in the **Directed Diffusion** paradigm [70]. Unlike SPIN where data is initially advertised by the source prior to being sent to the base station, in Directed Diffusion the base station floods the network with a message advising of its interest in a given item of data and if a node possesses such type of data, it will respond to the request. Directed Diffusion consists of three phases: interest propagation, data propagation and reinforcement (see Figure 2.3).



Source: A survey on routing protocols for WSN [7]

Figure 2.3: *The basic operation of Directed Diffusion protocol. (a) Interest propagation, (b) initial gradients setup, (c) data delivery along reinforced.*

First, the base station requests data by broadcasting *interests* – a list of attribute-value pairs such as the names of objects, durations, intervals, geographical area *etc.* The receiver of an interest packet caches it and forwards it to its neighbours if it is new. As the interest is propagated throughout the network, gradients are set up to facilitate data delivery to the base station. Each sensor that receives the interest sets up a gradient (toward the sensor nodes from which it receives the interest) which specifies both a direction to relay data and an attribute value which represents, in a sense, the importance or usefulness of a given link (*e.g.* it can be a data rate or expiration time) and which can be learned from the interest entries. When a receiver of the interest packet has data to send in response, it becomes a source and starts to send sensor measurements. An intermediate node forwards all incoming packets according to its gradients. This may lead to a significant overhead as identical packets are travelling over multiple, redundant paths. Hence, the best path is reinforced by simply disseminating the original interest with a higher value for the rate of data transmission attribute. An additional power saving may be obtained by aggregating data at the intermediate nodes based on the data's name and attribute-value pairs. Also, Directed Diffusion is scalable because nodes need only local information about network topology in order to process packets. Intanagonwiwat *et al.* claim that the reinforcement mechanism can adapt to the new topology resulting from a link or node failure [71]. Besides, the same experimental study shows that Directed Diffusion exhibits much better energy dissipation compared to flooding, while having good latency (for some sensor fields, its dissipated energy is only 60% that of flooding and at the same time every delivery has less than 20% additional average delay compared to flooding). The main difficulty with Directed Diffusion is the implementation of data aggregation which requires significant memory space which can add to the cost of a sensor node.

Rumor routing [18] is a variation of Directed Diffusion and its key idea is to limit the protocol overhead due to the flooding of the network with event queries. Generally Directed Diffusion disseminates interests to all nodes when there is no geographic criterion to diffuse tasks. However, in some cases there is only a little amount of data requested from the nodes and thus the use of flooding is unnecessary. It is then better to use an alternative approach, so called event flooding where the node which discovered an event disseminates the information about it (see Figure 2.4).

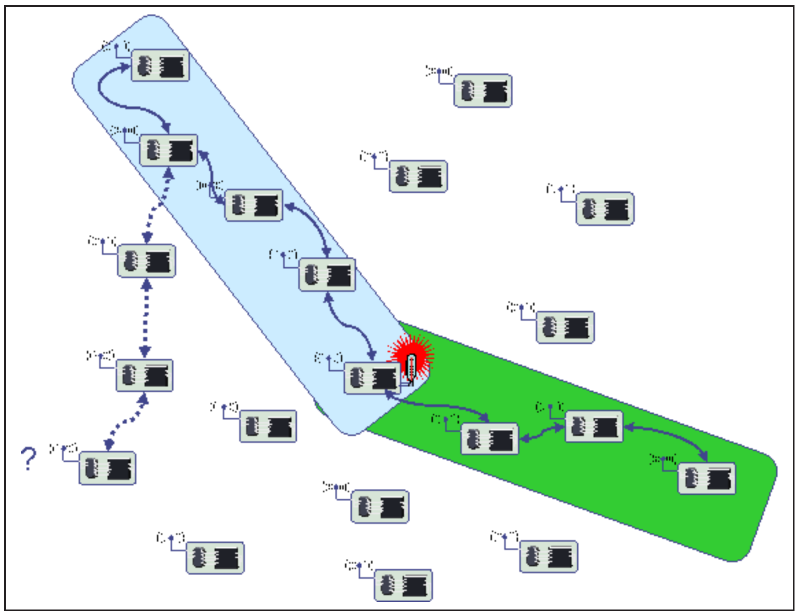


Figure 2.4: *Rumor routing: A node in the middle detects an event and sets up two event paths. A node (in the lower left corner, marked with "?") starts a query, propagating until it meets a preinstalled event path.*

Rumor routing combines both methods, query flooding and event flooding, in a random way. When a node detects an event, it generates one or several long lived packets called agents and lets them travel on a random path with related event information. Each visited node stores the routing information about the corresponding event. When a node tries to query for an event it also generates one or more agents. Once an agent reaches a node that has the information about a requested event, it will be routed to the event. Hence, there is no need to flood the whole network with event information, which reduces the protocol overhead. Simulation results have shown that rumor routing outperforms event flooding in terms of energy efficiency and can also handle node failure. The rationale behind rumor routing is the relative high probability that two random lines in a square intersect each other, which according to the authors is 69% and can attain 99.7% if five event paths are generated instead of just one. However, rumor routing is attractive only when the number of events is small. For a large number of events, the cost of maintaining agents and event tables in each nodes may not be paid back if there is not enough interest in those events from the base station.

Constrained Anisotropic Diffusion Routing (CADR) [25] strives to be a general form of Directed Diffusion and a complementary algorithm to the Information-Driven Sensor Query (IDSQ) [25] algorithm. CADR and IDSQ were especially designed for localisation and target tracking applications. IDSQ collects reports about events from sensors belonging to a local cluster. A normal cluster member simply waits for a query and delivers its local measurement when asked to do so by a cluster-head. The cluster-head successively selects a node for querying on the basis of a maximal incremental information gain and the node's position. The information utility measure is modelled using standard estimation theory. The application of IDSQ scheme is restricted to the local cluster. Hence, CADR can be applied when a query cannot be answered by nodes within a cluster. Here, a query is assumed to float through the network according to a set of information criteria. The main difference from Directed Diffusion is that CADR is a single path greedy algorithm that routes a query to its optimal destination using the local gradients (combining the information utility measure and the energy cost of communication with a neighbour) to maximise the information gain through the WSN. So upon receiving a query, the node evaluates an information/cost objective and forwards the packet based on the local information/cost gradient and end-user requirements. Simulation results show that CADR and IDSQ are more energy efficient than Directed Diffusion and provide lower packet latency. The main limitation of CADR is the lack of any mechanism to bypass network holes.

2.2.4.2 Flat protocols

In non-hierarchical protocols, each node typically plays the same role and sensor nodes collaborate to perform sensing tasks. Some of the protocols described in the previous section can fit into this category, however in this section I explore only node-centric algorithms.

Sequential Assignment Routing (SAR) [140] is a multi-path, table-driven and the first routing algorithm for WSNs which incorporates the notion of QoS into routing decisions. SAR performs three main tasks: path construction, packet forwarding over existing paths and path maintenance. To avoid single route failure, SAR creates trees outward from each neighbour of the base station until most nodes are part of several such trees. The created trees are used to construct multiple paths from base station to sensors. To reduce the multi-path induced overhead SAR reduces the reuse of the nodes for the same path to that part of a network where that really matters - near the base station. The nodes close to the base station are (often) those that are likely to fail first because of exhausted battery resources due to their processing of packets from distant nodes in addition to their own packets. Also the path construction process avoids nodes with low energy resources and without QoS guarantees. To deliver a packet to the base station, the packet source selects the path with the minimum value of a weighted QoS based metric. This metric is a product of the additive QoS metric (corresponding to delay over a path) and a weight coefficient associated with the priority level of the packet. The base station periodically initiates recomputation of paths to account for any changes in topology due to failures of nodes and also to mirror the evolution in available battery capacity of nodes. A handshake procedure based on a local path restoration scheme between adjacent nodes is used to recover from link failure.

In comparison to the minimum energy metric algorithm (which considers only the energy consumption of each packet without taking into account its priority), SAR offers lower power consumption. It ensures QoS, and features fault tolerance and easy recovery. However this comes at the price of a large overhead incurred in maintaining and updating large routing tables, especially in networks with many nodes.

SPEED [57] is another QoS routing protocol for WSNs which provides soft real-time end-to-end guarantees. Each node periodically updates its information about neighbours and their locations by using beacon packets. Upon reception of a packet, a node selects the next hop among the set of its neighbours that are closer to the final destination and have a retransmission delay (estimated on the basis of the elapsed time when an ACK is received from a neighbour in response to a transmitted data packet) smaller than a certain threshold t. The relay is chosen according to a discrete exponential distribution to favour the nodes with the fastest retransmission speed. If a potential relay candidate node exists, SPEED guarantees delivery of a packet within time d/t, with d being the distance between source and final destination. Furthermore, the stochastic selection process enhances traffic balance, and thus reduces congestion, since packets are dispersed over multiple routes. In the case when a suitable relay candidate cannot be found, the *relay ratio* of the node is checked, calculated by looking at the miss ratios of the neighbours of a node (the nodes that could not provide the desired speed). If this ratio is less than a randomly generated number between 0 and 1, the packet is dropped. SPEED provides also a back-pressure rerouting to prevent voids, when a node fails to forward the packet, and to eliminate congestion by sending a message back to the source nodes so that they will pursue new routes.

SPEED performs better than DSR and AODV in terms of miss ratio and end-to-end delay. Moreover, the greater energy efficiency of SPEED is due to the reduced control packet overhead and its mechanism of even traffic distribution.

Minimum Cost Forwarding Algorithm (MCFA) [154] is a lightweight routing protocol that exploits the fact that most data is travelling in the same direction, *i.e.* from sensors toward the fixed base station. Hence, MCFA does not use node ID nor maintain routing tables for every node. The protocol operates in two phases: a set up phase which determines the cost of transmission from each node to the base station and a forwarding phase where sensor samples are sent from nodes to the base station. The first phase is initiated by the base station which broadcasts a message with the cost set to zero, while every node initially sets its least cost to the base station to infinity (∞). Upon receiving this message, each node checks if the sum of the cost included in the message and the cost of the link from which the message was received is less than its current local estimate. If not, the packet is dropped. Otherwise, the local estimate and the estimate in the broadcast message are updated with the summation of these costs and packet is broadcasted. To increase the energy efficiency of this phase and reduce the number of broadcasts MCFA introduces a back-off mechanism. Instead of instantly resending the update message, a node waits $a * l_c$ units of time until transmission, where a is a constant determined through simulations and l_c is the cost of the link from which the message was received. At the end of this round, the cost value at every node captures the effect of delay, throughput and energy consumption from the node to the base station.

In the second phase, the source broadcasts measurements to its neighbours. The packet also carries two values: the minimum cost from the source to the base station and the total cost that it has consumed so far starting from the source to the current intermediate node. On receiving this message a node checks if it is on the least cost path between the source and the base station. This is so when the sum of the consumed cost (carried in the message) and the cost at this node matches the source's cost (also included in the message). If this is the case, it rebroadcasts the message to its neighbours. This process is repeated until the packet reaches the base station. Simulation results show that the cost values for each node obtained with MCFA are the same as with flooding. Thus, optimal forwarding is achieved with MCFA with the minimum number of advertisement messages. The average number of advertisement messages could be reduced by a factor of 50 using the back-off mechanism. The key problems of MCFA are the lack of a recovery mechanism from failure and the absence of topology update. The solutions to these problems are not so straightforward and because the frequent repetition of the first phase may be inefficient.

2.2.4.3 Hierarchical protocols

The main goal of hierarchical routing is to efficiently maintain the energy consumption of sensor nodes by involving them in multi-hop communication within a particular cluster and by performing data processing (aggregation or fusion) in order to decrease the number of messages transmitted to the base station.

Low-Energy Adaptive Clustering Hierarchy (LEACH) [61] is one of the first and most relevant clustered schemes for WSNs. It is a self organising protocol where nodes form local clusters with randomly selected cluster-heads. The cluster-head's objective is to organise the intra-cluster communication, collect data from subordinate nodes, aggregate it, and retransmit it to the base station. Because the cluster-heads dissipate more energy than other nodes, the authors of LEACH propose to rotate them periodically.

The operational time of LEACH is divided into two phases: a set-up phase and a steady phase. In the setup phase each node decides whether or not to become a cluster-head for the current round. Node n chooses a random number between 0 and 1. It becomes a cluster-head if the number is less than the threshold $T(n)$ calculated as

$$T(n) = \begin{cases} \frac{P}{1-P*(r mod \frac{1}{P})} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

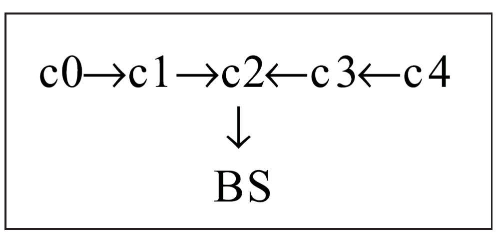
where P is the desired percentage of cluster-heads, r the current round and G is the set of nodes that have not yet been cluster-heads in the last $\frac{1}{P}$ rounds. With this approach every node becomes a cluster-head once in the $\frac{1}{P}$ rounds. After the cluster-heads are elected they advertise their status to all nodes. Then the sensors select the closest cluster-head and inform it about their decision. Afterwards the cluster-head assigns time slots for transmission to the subordinate sensors and distributes the schedule among them. This schedule is used for communication in the steady phase. After following the TDMA schedule for a certain time, nodes repeat the set-up phase and so change the transmission topology.

The LEACH protocol is attractive for many reasons. It is distributed, which facilitates the scalability of the protocol. The small number of cluster-heads and their periodic rotation extend the lifetime of the system. The performance of LEACH compares favorably with other WSNs routing protocols [7, 61, 72]. However the original version of LEACH had some drawbacks. It uses single-hop routing where each node transmits to the cluster-head and then to the base station. This approach drains a significant amount of energy from the cluster-head and makes the protocol unsuitable for large area networks. The process of cluster-head election does not consider the remaining energy at the nodes. Also, the requirement of node synchronisation may be very energy demanding without careful design.

Some of these issues were addressed in later versions of the protocol. Heinezlman *et al.* propose a variant of LEACH, called LEACH-centralised (LEACH-C) [62]. It is characterised by a centralised cluster-head election which prevents an unfair distribution of cluster-heads. LEACH-C is more energy efficient than the former protocol but the centralised cluster formation approach makes it impractical for large size networks.

Mhatre *et al.* compare heterogeneous and homogeneous clustered networks in terms of the manufacturing cost of hardware and the sum of the energy costs [103]. They propose a multi-hop variant of LEACH called M-LEACH. In this protocol a node needs more than one hop to reach the cluster-head. M-LEACH out-performs LEACH for a large propagation loss index ($k > 2$).

Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [91] and Hierarchical-PEGASIS [90] are both claimed by their authors to be an enhancement over LEACH. PEGASIS addresses a scenario where homogeneous sensor nodes have to transmit their local measurements to a known base station, once per given round (rounds are synchronised by *e.g.* high-powered beacon signals from the base station). In order to extend the network lifetime, nodes need only to communicate with their closest neighbours and form communication chains for every round. Nodes take turns to be a chain leader and to transmit to the base station, balancing the power drain uniformly over all nodes. Because each node has global knowledge of the network, the chain can be constructed easily by using a greedy algorithm. The chain leader uses a simple control token passing approach to inform the root of the chain about the start of next transmission round. It passes the token to the node farthest away from the base station. The chain grows from one end only and the next hop to be added is the as-yet unselected node nearest to the current end node. The gathered data moves across the chain from node to node and at each hop it is aggregated. When the token reaches the leader it either forwards the token to a node which has not sent its data in this round yet or transmits the collected data to the base station. The operation of PEGASIS is shown in Figure 2.5.



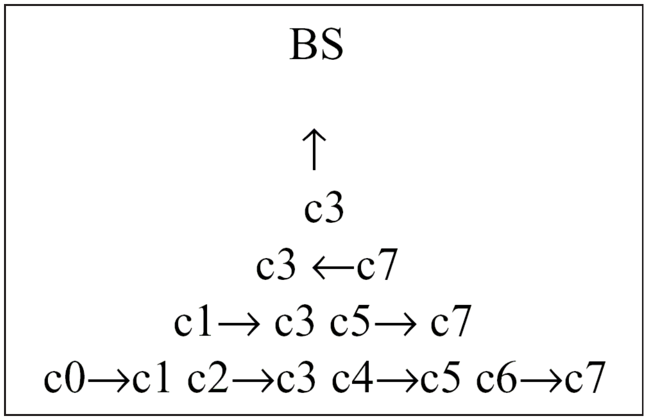
Source: PEGASIS: Power-Efficient Gathering in Sensor Information Systems [91]

Figure 2.5: *Chaining in PEGASIS: node c2 is the leader, and it gathers data from the chain beginning with node c0. After node c2 receives data from node c1, it passes the token to node c4.*

To locate the closest neighbour, each node uses the received signal strength to measure the distance to all neighbouring nodes and then adjusts its transmitted signal strength so that only one node can be heard. Simulation results show that PEGASIS can extend network lifetime to twice that available using the LEACH protocol. This energy efficiency improvement is mainly due to the decrease in the number of transmissions by using data aggregation and to the elimination of the overhead caused by dynamic cluster formation in LEACH. However, PEGASIS suffers from excessive delay for distant nodes on the chain. Furthermore it does not scale well and is not suitable for large networks because PEGASIS requires global network knowledge at each node which is not only difficult to obtain but also demanding of memory space. Also, PEGASIS assumes that each node can communicate with the base station directly. Finally, PEGASIS does not provide any mechanism to deal with topology changes due to node failure.

Hierarchical-PEGASIS is an extension to PEGASIS which addresses the problem of packet delay incurred by packets during transmission to the base station. To reduce it, the protocol uses a simple simultaneous transmission strategy where firstly direct neighbours send data to each other, *e.g.* odd numbered nodes send to even-numbered nodes. Then, in the next step, only those nodes that were receiving data in the previous step remain active and one half of them send their aggregated data to their neighbours. These steps repeat until the aggregated data has arrived, as a single transmission, at the current leader (see Figure 2.6).

To avoid collisions and possible signal interference among the sensors during these simultaneous transmissions, two solutions have been investigated. The first approach uses signal coding, *e.g.* CDMA. This method ensures data can be transmitted in parallel and reduces the delay significantly. Since the tree is balanced, the delay will be in $O(\log N)$ where N is the number of nodes. When CDMA is not available, the second approach is applied where only spatially separated nodes are allowed to transmit at the same time. This method reduces packet delay significantly in comparison to LEACH and the original PEGASIS protocol and it has been shown that in terms of the *energy*delay* metric it performs better than the



Source: Data Gathering in Sensor Networks using the Energy Delay Metric [90]

Figure 2.6: *Data gathering in a chain based binary scheme with Hierarchical-PEGASIS: Since round leader (node c3) is in position 3 (counting from 0) on the chain, all nodes in an even position send to their right neighbour. Nodes that are receiving at each level rise to next level in the hierarchy. Now at the next level, node c3 is still in an odd position (1). Again all nodes in an even position will aggregate its data with its received data and send to their right. At the third level, node c3 is not in an odd position, so node c7 will aggregate its data and transmit to c3. Finally, node c3 will combine its current data with that received from c7 and transmit the message to the base station..*

regular PEGASIS by a factor of about 60. Although hierarchical-PEGASIS solves the delay problem in the original PEGASIS protocol, it does not address the other limitations of this algorithm.

Threshold-Sensitive Energy Efficient Sensor Network Protocol (TEEN) and Adaptive Periodic TEEN (APTEEN) [94, 95] are both cluster based protocols and are similar to LEACH except that sensor nodes do not have to send at a fixed rate. The protocols were proposed for time-critical applications and they employ the cluster formation strategy of LEACH but adopt a different strategy in the data transmission phase. Transmission of data is very energy demanding, and so to extend the network lifetime nodes transmit only those measurements which fulfill some predefined requirements. After the formation of clusters, cluster-heads send cluster members a hard threshold (HT), which is the threshold value of the sensed attribute, and a soft threshold (ST), which is a small change in the value of the sensed attribute that triggers the node to power on its transmitter and send measured data. So, a sensor node senses the medium continuously and whenever the sensed attribute is greater than HT, the data is reported to the cluster-head. Thus, the HT allows the nodes to control the number of transmissions since only the values which are in the range of interest are sent. Once the sensed attribute is at or beyond the HT, it reports data only when the value of that attribute changes by an amount equal to or greater than the ST. This further reduces the number of transmissions. The user can control the tradeoff between data accuracy (which is better for small values of ST) and energy efficiency (which increases with the value of ST). However, TEEN is not suitable for applications which require periodical reporting of data since the base station may not get any data at all if the thresholds are not reached or the thresholds were not delivered to a node.

APTEEN is an extension to TEEN which aims at both capturing data collections and reacting to time critical events. The cluster formation process is the same as in TEEN, as is the aggregation of data by cluster-heads. On the other hand, APTEEN is a hybrid protocol that changes its parameters (*e.g.* periodicity or HT and ST values) according to user needs and the application type. In APTEEN, the cluster-heads distribute more parameters to cluster members. These are:

- Attributes: a set of physical parameters about which the user is interested in obtaining information
- Thresholds: the same HT and ST as used in TEEN
- Schedule: a TDMA schedule assigning a slot for every cluster member
- Count Time: the maximum time period separating two successive reports sent by a node

A node using the APTEEN protocol reports sensor measurements in a similar way to the TEEN scheme. However, nodes operate according to the schedule sent by the cluster-heads. Also, if a node does not have any data to send for a period *count time*, it is forced to sense and transmit the data. The main enhancement of APTEEN is that it offers more flexibility by allowing the user to set the *count time* interval and also the energy consumption can be controlled by adapting the *count time* and threshold values. The flexibility advantage is obtained at the price of additional protocol complexity.

Simulation results have shown that TEEN and APTEEN exhibit better energy efficiency and network lifetime than LEACH. Of the three protocols TEEN achieves the best energy saving since it decreases the number of transmissions. The main limitations of TEEN and APTEEN are the overhead and complexity involved in forming clusters in multiple levels, implementing threshold-based functions and dealing with attribute-based naming of queries.

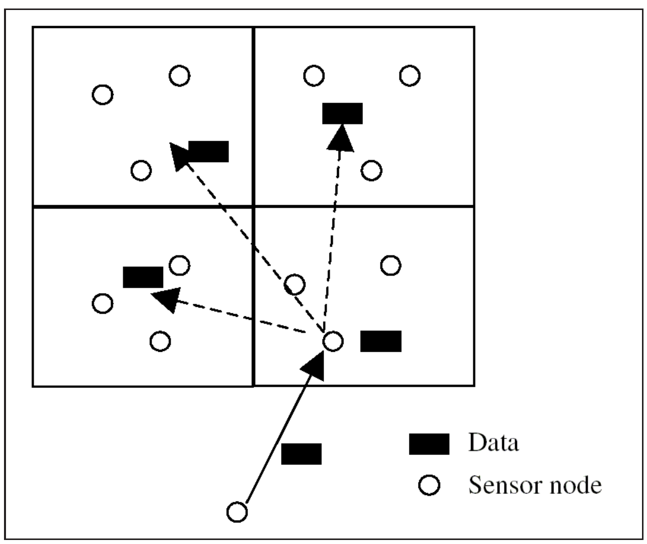
2.2.4.4 Location-based protocols

Geographic based routing protocols are of benefit with applications where the base station addresses nodes by physical location (*e.g.* any node in a given region). In this case there is no need to use node resources to determine each node coordinates uniquely for the purpose of routing. The key benefit of geographic routing is a much simplified protocol with significantly smaller or even nonexisting routing tables as physical location carries implicit information concerning the next hop for a packet.

Geographic and Energy Aware Routing (GEAR) protocol [157] is a modification of the directed diffusion paradigm. It forwards queries to appropriate regions according to the geographic information often included in the interest packets. This in consequence, restricts the number of interests in Directed Diffusion by only considering a certain region rather than flooding the whole network. GEAR is composed of two phases:

- Forwarding the interest packets in the direction of the target region: Upon receiving a packet, a node selects a next hop which corresponds to the lowest so-called *learned cost*. The *learned cost* is a refinement of the *estimated cost* (which is proportional to a linear combination of residual energy and distance to the destination) that takes into account the cost of routing around holes[13] in the network. If there is no such neighbour, it means there is a hole and then one of the neighbours is picked to forward the packet based on the learned cost function.

- Forwarding the interest packets within the region: When the packet reaches the region, GEAR uses either recursive geographic forwarding or restricted blind flooding to disseminate the packet within that region. Recursive forwarding is more energy efficient than restricted flooding in high-density networks and it applies GEAR to send messages to four subregions in the routing region, which repeats until the region has a single node inside it (as shown in Figure 2.7).



Source: A survey on routing protocols for WSN [7]

Figure 2.7: *Recursive geographic forwarding in GEAR.*

GEAR outperforms the popular geographic routing protocol GPSR [81] in terms of energy efficiency and packet delivery. Experimental results show that GEAR delivers 60-70% more packets than GPSR for an uneven traffic distribution and 25-35% in other scenarios. The most important limitation of GEAR is that it performs well only in static networks.

[13]A routing hole of the geographic routing protocol corresponds to a node whose cost function is a local minimum; no neighbour nodes are closer to the destination than the node itself.

Geographic Adaptive Fidelity (GAF) [152] is an energy-aware location based routing protocol designed initially for ad hoc networks, but that may be applied to WSNs as well. GAF conserves energy by exploiting redundancy in the network and turning off unnecessary nodes in in the network without affecting the level of routing and environment monitoring fidelity. To do this, GAF divides the covered area into rectangles that are small enough such that any node in one rectangle can communicate with any other node in an adjacent rectangle. Nodes belonging to the same grid area are considered equivalent and some of them can be in a sleeping state for a certain period of time in order to save energy. The size of the grid rectangle depends on the transmission range and equivalent nodes can be identified in the following way. The critical positions are those that are at diametrically opposed corners of two rectangles; two such nodes should be able to communicate with each other at the maximum radio range R (see Figure 2.8).

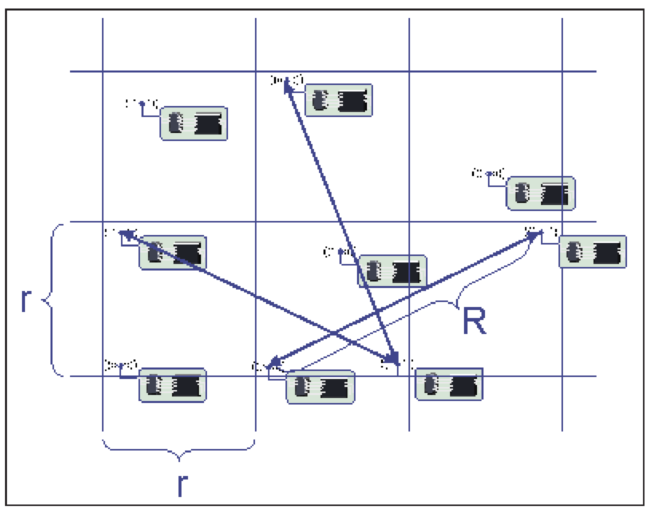


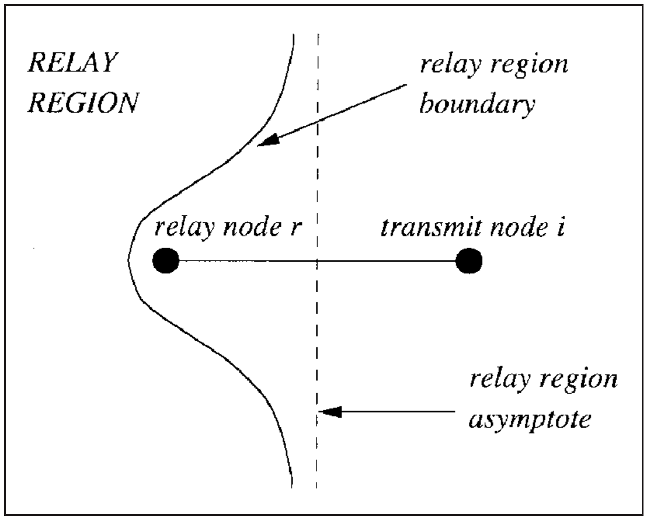
Figure 2.8: *Relationship between maximum radio range R and rectangle length r in the GAF protocol.*

The distance between two such critical nodes is $\sqrt{r^2 + (2r)^2}$. As this distance has to be smaller than R, it follows that $r < R/\sqrt{5}$. Since the nodes know their location, they can easily construct such equivalency rectangles, determine what nodes are in their own rectangle and collaboratively establish a sleeping pattern. There are three states defined in GAF: discovery (for determining neighbours in the grid), active (reflecting participation in routing) and sleep (when the radio is turned off).

GAF is implemented both for non-mobility (GAF-basic) and mobility (GAF-mobility adaptation) of nodes. In the latter case, each node in the grid estimates its time of departure from the grid and transmits this to neighbours. The sleeping neighbours adjust their sleeping time accordingly in order to maintain routing fidelity. Before the departure time of the active node, sleeping nodes wake up and one of them becomes active. A comparison of GAF with other ad hoc networks routing protocols proves its better energy efficiency (between 40 and 60%) and in consequence longer lifetime. Also, it performs at least as well as the compared protocols in terms of latency and packet loss. However, GAF may be difficult to implement in WSNs because of the required location awareness of every node. The authors assume that nodes can learn their coordinates by using GPS receivers, which is impractical with current WSNs technology.

Minimum Energy Communication Network (MECN) [128] is another location based routing protocol that constructs power-optimised paths between a set of source nodes and the base station. The MECN is based on the concept of a so-called relay region which is identified for every node. Given a pair of nodes i (transmit) and r (relay), the relay region is an area which covers all potential locations of a destination which should receive packets through node r from node i and not directly from node i in order to make the communication energy efficient (see Figure 2.9).

MECN has two phases. First, a node identifies its enclosure which is the region containing nodes with which the given node should directly communicate. The enclosure graph consists of all the enclosures of each transmit node and contains globally optimal links in terms of energy consumption. As a result each node obtains a reduced set of immediate neighbours through which transmission is more energy efficient than by direct communication. Thus in a second phase optimal routes can be constructed in a more power-efficient way, since communicating di-



Source: Minimum energy mobile wireless networks [128]

Figure 2.9: *Relay region of node i with node r as possible relay.*

rectly only with neighbours in the enclosure graph requires less power than communicating directly with all neighbours. Optimal routes are found by applying the distributed Bellman-Ford shortest-path algorithm with power consumption as the cost metric. MECN is self-reconfiguring and thus can dynamically adapt to node failure or the deployment of new sensors. The reconfiguration can be performed by executing the first phase of the protocol between two successive wake-ups of nodes, and the minimum cost links are updated by identifying leaving or newly joining nodes. As with GAF, the main drawback of MECN is that location is estimated with GPS which is not suitable for WSNs.

2.3 Cross-layer design approach

The concept of WSNs has many applications in different areas such as the medical, environmental, civilian and military domains. Apart from novel opportunities, this technology brings also new design challenges mainly due to the distributed nature of WSNs and the limited resources of nodes. To overcome these problems, many novel architectures and approaches have been proposed that implicitly and explicitly violate the rules of strictly layered design, cutting across traditional layer boundaries. These many different solutions and motivations for cross-layer design in WSNs are presented below.

2.3.1 Motivations for layered communication system design

The traditional approach to communication protocol design is to use layering *i.e.* individual protocols are stacked on top of one another and each layer only uses functions of the layer directly below it. The layered architecture was borrowed from computer systems where it has been used for a long time, *e.g.* in the von Neumann architecture [110] - a computer design model consisting of memory, control unit, input-output devices, and arithmetic and logical unit, which is at the heart of computer systems. There are three important advantages of using a layered architecture. First, it provides the abstractions necessary for designers to understand the overall system. Hence it allows a complex problem to be broken into smaller, more manageable pieces which are simpler to solve. It also accelerates the process of both design and implementation by enabling a parallelisation of efforts. Second, because the specification of a layer does not consider its implementation, the implementation details of a layer are hidden from other layers. Thus, individual modules can be upgraded without disturbing the rest of the stack which would stifle the further development and proliferation of the system. Third, many upper layers can reuse the functionalities provided by lower layers. This simplifies the implementation of complex systems (*e.g.* a server application) since a designer need only develop the upper layers and can reuse other modules (*e.g.* network functionalities).

2.3.2 Wired networks design approach

The layered approach was applied in the Open System Interconnect (OSI) reference model. It defines seven layers of network stack (physical, data link, network, transport, session, presentation, and application layer). Communication between nonadjacent layers is not allowed, whereas adjacent layers communicate through static interfaces, independent of the individual network constraints and applications. Although this layered approach has provided many advantages as previously described, it suffers from some limitations. For example, the fact that the lower layer presents only a service interface to an adjacent upper layer and hides any other information, can lead to poor performance [27]. To avoid this penalty, the protocol can reveal implementation information that would normally be hidden behind a layer boundary in situations where the performance of a protocol can be optimised by sharing of this data between layers. For example, consider a flow control protocol (*e.g.* TCP) that is responsible for informing a source when it thinks that the network is overloaded. In a widely used heuristic, to discover an overload the receiver measures the packet loss and it can throttle the source in case of a packet arrival failure. Usually this flow control protocol is layered above the protocol that is actually responsible for data transfer and so is unaware how packets are transferred across the network. If the end-system communicates over a wireless lossy link, so that most lost packets are corrupted because of link errors, the flow control protocol assumes that network is congested, and throttles a source even when there is no need to do so. The flow control protocol could distinguish between the link and congestive losses if information about packet loss on a link was available to the higher (flow control) layer from the lower (data transfer) layer. This, however, violates layering, because the flow control layer would then know about details of data transfer over its local link but the benefit resulting from such cross-layer information sharing is reflected in better performance of the flow control protocol.

2.3.3 Wireless networks design approach

Although cross-layer design sometimes can lead to better performance of protocols, the vast majority of communication algorithms for wired networks conforms to a strictly layered architecture scheme. In contrast, the cross-layer approach has gained significant popularity in wireless networks because the wireless medium allows modalities of communication that are not possible in wired networks. This is mainly because there is no concept of a "link" in wireless networks and transmission is a spatio-temporal footprint of radio energy [82]. Hence, cross-layer design approach can be applied to support wired networking protocols in wireless systems (*e.g.* TCP over wireless [10]), to optimise handover mechanisms by physical layer information in cellular networks [48] or for efficiency reasons [28].

2.3.4 Reasons for using cross-layers design in WSN

The cross-layer approach has been applied many times in the development of communication protocols for WSNs [55, 65, 84, 136, 151]. We can distinguish numerous factors motivating cross-layer design:

The necessity of a lightweight protocol in WSNs because of **constrained resources** available at each node motivates the use of cross-layer approach. Thus, the prevalent trend in building a more flexible communication architecture in WSNs is to use a component model where relatively large, monolithic layers are broken up into small, self-contained "components", "building blocks" or "modules". Every component fulfills only one, well-defined task, *e.g.* packets scheduling, and interact with other modules over clear interfaces. The main difference compared to the layered architecture is that these interactions are not necessarily vertical between immediate neighbours, but can involve any components. A popular example for an operating system following this approach is TinyOS [64], which is widely used by the WSN research community. It supports modularity with the components arranged hierarchically from low-level components close to the hardware to high-level components composing the actual application. The components exchange information by the mean of predefined interfaces and it is possible that non-adjacent components in a hierarchy are connected by an interface.

The cross-layer performance aspects can be simplified with cross-layer design approach. WSN nodes must manage several performance aspects, such as system management, power management, security, that cut across traditional layers. For example, physical, medium access and routing decisions have a significant impact on power consumption, and the joint consideration of them can yield more efficient power consumption. The traditional strictly layered architectures do not enable sufficient interaction among layers to make joint decisions to optimise these cross-layer aspects.

Mobility introduces an additional challenge for WSN design. The fact that nodes have limited energy resources makes frequent adaptations of node behavior in response to the changing network topology very difficult. Mobility causes changes for the physical layer (*e.g.* interference levels), the data link layer (*e.g.* link schedules), the routing layer (*e.g.* new neighbours), and the transport layer (*e.g.* connection timeouts). Hence, a cross-layer design enhances a node's ability to manage its resources in mobile environments.

Wireless links are more susceptible than wired links to interference variations and channel errors. One classic example is the TCP congestion control problem in wireless links already discussed (see section 2.3.2). Providing higher layers with awareness of the wireless link status enables nodes to adapt their configuration better to physical layer properties. For example, a routing protocol that detects a drop in quality of a particular wireless link can create a new route to divert traffic to another wireless link.

Existing **interlayer dependencies** motivate cross-layer design for WSNs. For example, the operation of data link layer is closely related with the performance of the physical layer. If provided with current channel conditions, the data link layer can adapt error control mechanisms in a dynamic manner, thereby improving throughput.

2.3.5 Potential drawbacks of cross-layer design

The possibility for cross-layer information sharing holds great promise for protocol optimisation, but some researchers have argued that it can have a potentially negative effect on system performance. Kawadia *et al.* [82] emphasize the tradeoff between performance and architecture: while cross-layer design offers short-term performance benefits for a particular system over traditional architectures, it also limits the modularity and interoperability offered by architectures. We can distinguish the following potential drawbacks of cross-layer design:

Unstructured code: The implementation of several cross-layer design optimisations within a system may lead to spaghetti-like code that is unstructured and thus difficult to maintain. Further design developments may then be stifled since it will be hard to modify or upgrade existing systems. Finally, the unstructured code could eventually lead to an increase in per-unit cost. All of these problems can appear as a long term consequence of cross-layer design considering only short term gain.

Multiple interactions: Cross-layer design opens the floodgate of information flow across layers, raising concerns on multiple, sometimes subtle, interactions among existing layers. In some cases cross-layer approach can create “loops”, and it is well known from control theory that stability then becomes a paramount issue. Thus, a major challenge of cross-layer design is the clear definition and exploration of the possible dependencies and interactions among the system processes at different layers.

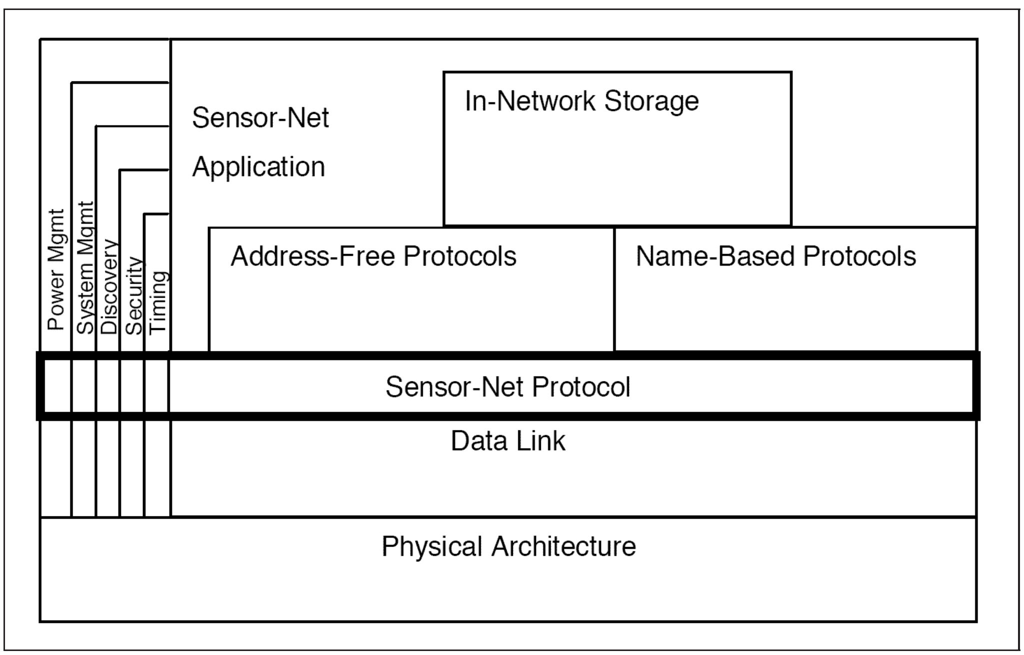
Holistic perspective: One of the advantages of layered design approach is that the individual modules can be upgraded without disturbing the rest of the system. The cross-layer design enhances dependencies among other system process and thus designers must evaluate the impact of their design with a holistic view.

Particularly, they must consider the long-term development and innovation considerations.

2.3.6 Cross-layer architectures

Cross-layer architectures describe the system design from a holistic point of view. The concentrate on how to interface or integrate layers of the communication stack. There have been several cross-layer architectures proposed for WSNs. Among these, we can identify the prevailing examples:

The Sensor-Net Protocol (SP) [34] architecture provides the guidelines for establishing a sensor network architecture that enables interoperability among different layered components. The authors propose to insert an abstraction layer, called Sensor-Net Protocol (SP), whose role is similar to that of IP in the Internet (as shown in Figure 2.10).



Source: Sensor-Net Protocol [34]

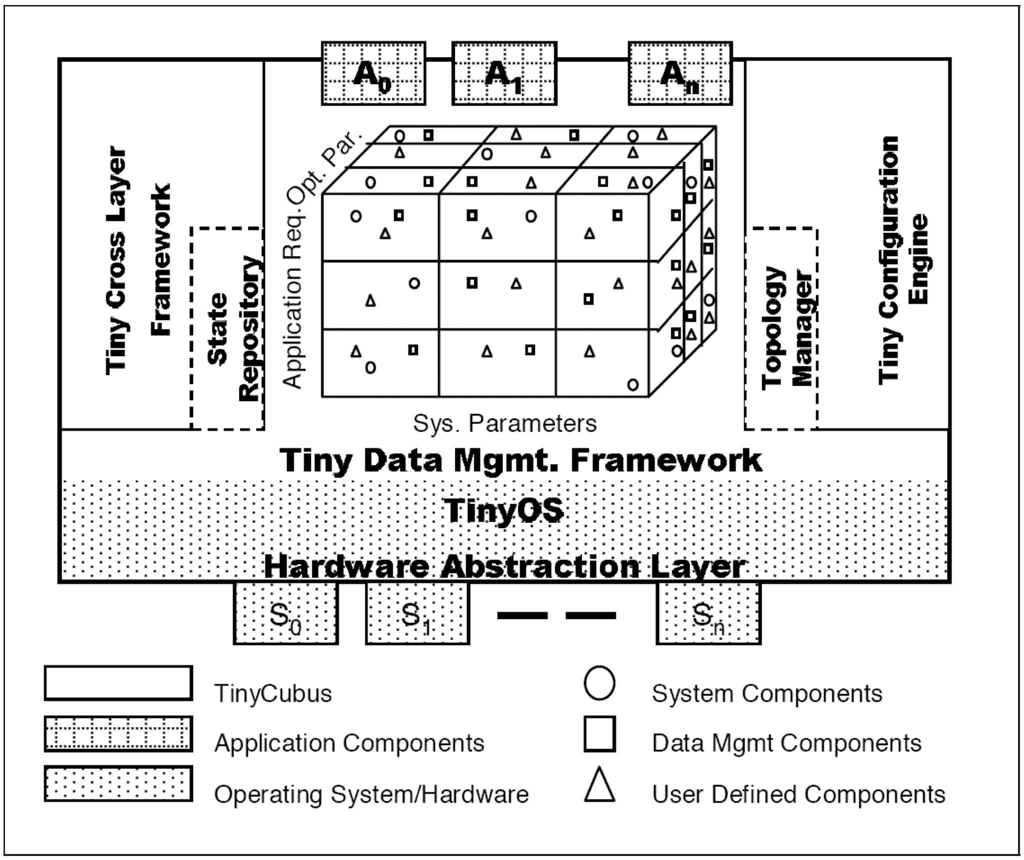
Figure 2.10: *SP architecture.*

However, SP sits between the network layer and data link layer because packet processing potentially occurs at each hop and not just at the end points. SP serves as "narrow waist" of the architecture and allows multiple network protocols and link technologies to co-exist. All higher and lower layer protocols and services need only interface with the SP protocol. The architecture also introduces cross-layer visibility and management of several aspects, such as system management, power management, discovery, security and timing. In particular, the SP architecture specifies that all layers in the system should have access to these services. The SP architecture is platform independent and can be implemented in any OS.

TinyCubus [98] is a cross-layer framework for sensor networks based on TinyOS that aims at providing the necessary infrastructure to cope with the complexity of such systems. It has three main elements:

- *cross-layer framework*, which provides a generic interface to support parameterisation of components using cross-layer interactions – this element also has the *state repository*, which stores all relevant parameters for cross-layer access;
- *data management framework*, which provides a standard set of data management and system components and chooses the best component set based on three dimensions: system parameters (*e.g.* node resources, sensor density, or mobility), application requirements (*i.e.* QoS metrics such as reliability or delay), and optimisation parameters (*e.g.* energy efficiency, communication latency, or bandwidth);
- *configuration engine*, which deals with the task of distribution and installation of code in the network – this element contains the *topology manager* which executes the *role assignment algorithm* assigning a role to each node, such as SOURCE, AGGREGATOR, base station or CLUSTER HEAD (see Figure 2.11).

Although the experiments shows that this algorithm is able to reduce the number of messages sent to nodes which need update information [98], it is based

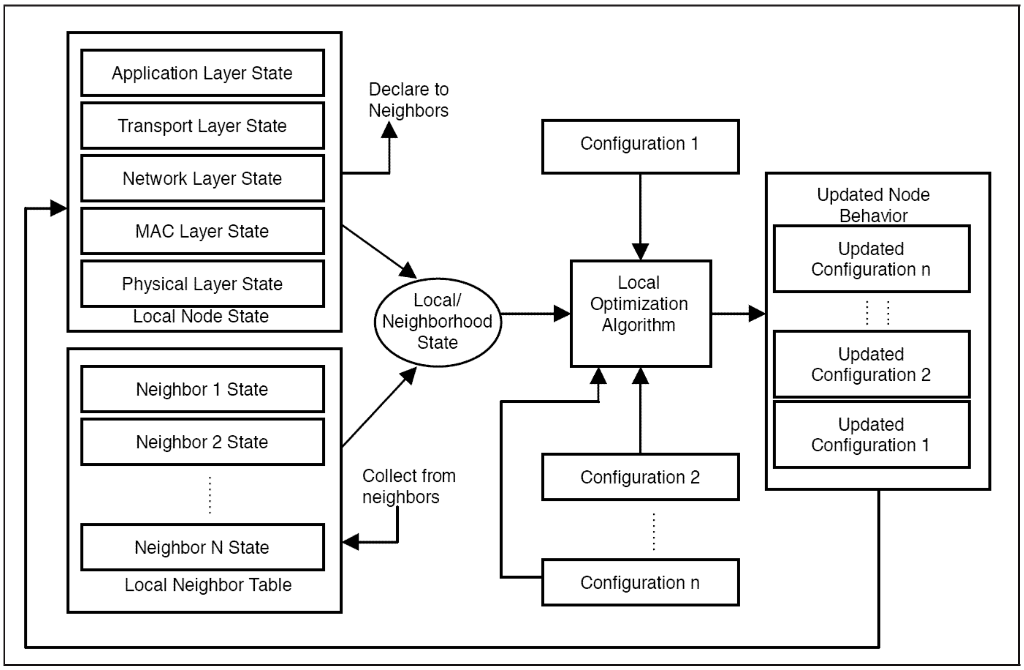


Source: TinyCubus [98]

Figure 2.11: *TinyCubus architecture.*

on a theoretical assumption that the WSN is characterised by a structured deployment.

Jurdak [78] describe a cross-layer optimisation framework for both ad hoc and sensor networks that advocates full visibility of relevant state information among communication layers. It specifies both interlayer and internode interactions with many details (as shown in Figure 2.12).



Source: Modeling and Optimization of Ad Hoc and Sensor Networks [78]

Figure 2.12: *Jurdak's cross-layer optimisation framework.*

The framework's state definition provides the network designer with flexibility in specifying the relevant state variables, to ensure that the framework is tunable to the performance requirements of different applications. For example, an ad hoc network for video transfer requires throughput and delay guarantees. In contrast, long term monitoring sensor networks require energy efficient behavior. These performance issues cut across layers, similar to the power management and system management aspects in the SP architecture, and they require internode collaboration.

Each node can declare its state information to its direct neighbours, and nodes maintain a state table of neighbouring nodes, which is denoted as the local neighbour state. The combination of the local and neighbour state at each node comprises the overall node state. The framework enables each node to use its local and neighbourhood information to adapt its routing and MAC layer behavior.

2.3.7 Examples of applied cross-layer approach

We can distinguish two main categories of the cross-layer design approach in WSNs. In the former type, called information sharing, adjacent or non-adjacent layers can share information through a new interface. Also, in this category a cross-layer architecture may introduce comprehensive state variables that are accessible throughout the communication stack [34]. The latter category, called design coupling, consists of partially or completely integrating the functionality of adjacent layers [32]. First, I present examples belonging to the former category and then I describe the solutions from the latter set.

2.3.7.1 Information Sharing Approaches

Information sharing approaches preserve the overall layered architecture while providing additional functionalities for interlayer interaction, in an attempt to balance generality and performance. These methods generally adopt a more functional and architectural perspective of cross-layer design than traditional layered solutions through consideration of the practical internode and interlayer interactions to support cross-layer optimisation algorithms.

Sichitiu's approach [136] is designed for data gathering applications characterised by stationary nodes and periodic, long-lived, and predictable traffic flow. The author proposes a deterministic schedule-based strategy that relies on sleep modes for promoting energy efficiency in sensor networks. It relies on the close coupling of the MAC and network layer for determining optimal schedules. This method defines two phases for each flow in the network:

- Steady state phase: Nodes remain most of their time in steady state phase, during which nodes perform tasks according to a fixed schedule table. The schedule table can have three types of entry: sample (corresponding to sampling a sensor), transmit (corresponding to packet transmission) or receive (corresponding to packet reception).

- Setup phase: An unexpected event *e.g.* node failures, battery depletion, or a change in the network objective may cause a flow to enter a short-lived setup and configuration phase, before going back to the steady-state phase. The setup phase is split into two steps: route selection, and route setup. The underlying routing protocol, which is left unspecified in this approach, handles the route selection step. The route setup step involves sending probe messages that can find appropriate schedules for the data transmission and reception on the links of the selected path. The route setup probes use a generic RTS/CTS MAC layer. The route probe message schedules transmit and receive actions at each intermediate node along the path, until it reaches the base station. The intermediate nodes store the schedule in a temporary table, until they receive an acknowledgement of the schedule from the data base station. The acknowledgement takes the reverse path to reach the original sender.

The main advantage is the realistic power consumption model for sensor nodes, which takes into account the transceiver wake-up power. The limited applicability of this approach to only periodic traffic flow is an important drawback. Also, the approach has limited scalability since it requires synchronisation of nodes and large storage space for maintaining active path information.

Jurdak's framework [78] defines a general structure for implementing optimisations and it enables the use of any optimisation algorithm to modify node configurations. The framework can support different applications, quality goals, and communication technologies depending on its adaptation to specific applications. One of the scenarios described by Jurdak is monitoring an event-driven RF sensor network test-bed. The considered application is a long-term monitoring deployment in which the sensor nodes periodically send a sensor sample to a single base station. Due to the long-term requirement of the application, the main goal of the information sharing approach is to promote energy efficiency through adaptive listening modes and to extend network lifetime through load balancing. To achieve this, the protocol duty cycles the nodes and uses (Berkeley Media Access Control) B-MAC [120] to periodically check the channel for activity. The proposed approach uses the Adaptive Low Power Listening (ALPL) method which runs locally at each node to enable the nodes to adapt their check interval according to their current state. It locally adjusts the MAC layer listening mode and the routing cost of neighbours. Test results have shown that the framework offers significant global energy savings compared to traditional layered solutions because it effectively balances the load in this scenario.

2.3.7.2 Design Coupling Approaches

Cross-layer approaches based on design coupling of adjacent layers disregard layer boundaries and consider the network mechanism as an integral block that optimises certain performance metrics. Typically, techniques applying that design method are focused on algorithmic design without taking into account functional or architectural issues.

Madan [93] presents a cross-layer optimisation approach whose goal is to maximise network lifetime, defined as the time until the first depletion of energy resources of any node. To satisfy the network lifetime constraint, the method adjusts transmission power, transmission rate, routes and link schedules. The author targets applications where every node generates data with a fixed rate, so he proposes that nodes communicate according to a TDMA schedule. He also advocates the use of multihopping to reduce transmission power consumption and load balancing to shift traffic load away from hot spots. Initially, the algorithm starts with a feasible suboptimal schedule. It determines the optimal transmission rates and powers for this schedule. In the next step it checks if the schedule is feasible, in which case it disables links with a SNR close to 1. Then, it allocates an additional time slot to a link with the maximum average power. This operation repeats until it converges to the optimal solution. At every iteration, the algorithm terminates in the case of a repeated schedule, an infeasible schedule, or an infeasible SNR. Repeated checking of these constraints ensures that the schedule determined at the end of each iteration is feasible. The key benefit of this approach is its potential for distributed computation, although the mechanisms for distributed implementation are not clear. The goal of this approach which is to maximise the sensor network lifetime makes this technique highly practical for WSNs. However, the main drawback of this approach is its purely algorithmic nature that does not specify implementation details for mapping the algorithm to interlayer or internode interactions in existing architectures.

Cui [33] describes a cross-layer design approach that targets energy efficiency in small-scale sensor networks. The author argues that the hardware power consumption caused by data processing and sensing, may equal or even overweigh communication power consumption when communication links span short distances. Hence, he considers both the hardware power and the communication power as the causes of power consumption to jointly determine link schedules, communication paths, M-ary Quadrature Amplitude Modulation (MQAM) modulation rates and transmit powers that yield better energy efficiency in the network. To study energy consumption of a node Cui uses an energy model that considers two power states: active (transmission or reception of data), and sleep (all circuits are off). It also explores the energy-delay tradeoffs, in particular the queuing and transmission delay which are the main elements of total packet delay. The author argues that in a TDMA tree topology, scheduling a node's incoming links before its outgoing links on a particular route minimises packet delay.

A key contribution of Cui's approach is the joint consideration of link schedules, modulation rates, transmission power, and routes. Also the power consumption modeling is more accurate than usually used for theoretical analysis of the protocol performance because it considers hardware power consumption in addition to transmission power consumption. The disadvantage of this approach lies in its requirement of periodic sensor sampling by every node which limits the applicability of the method and is not suitable for dynamic data reporting scenarios.

2.4 Related topics

In this section different issues related to routing in WSNs are presented. Energy efficiency is one of the major objectives when designing routing protocol for WSNs. Energy is mainly drained from a node by the transceiver and the communication cost mainly depends on the transmission distance. Hence, the need to understand the wireless propagation model and related to it, so called transition distance. Also, node's energy can be saved by duty cycling the nodes. Because it requires time synchronisation of the nodes, this section describes difficulties and other advantages of achieving clocks' synchronisation.

2.4.1 Wireless propagation model

Most commercially available WSN transceivers enable adjustment of transmitted power and hence variation of communication distance. This transmission range has a significant influence on routing topology and energy efficiency. Thus, we describe the log-normal shadowing path loss model which enables estimation of communication distance.

2.4.1.1 Log distance path loss model

Some of the earlier path-loss empirical measurements, carried by Okumara *et al.* [153] indicate that average received signal power decreases logarithmically with distance in both indoor and outdoor environments. Okmura's measurement data were transformed by Hata [56] into parametric formulas. Hence the the mean path loss $\overline{PL}(d)$ as a function of distance d between transmitter and receiver is proportional to an nth-power of d relative to a reference distance d_o:

$$\overline{PL}(d) \propto \left(\frac{d}{d_o}\right)^n \tag{2.2}$$

Because the $\overline{PL}(d)$ is often expressed in decibels, so the equation 2.2 becomes:

$$\overline{PL(d)}_{\,dB} = PL(d_o) + 10\,\eta\,\log_{10}\left(\frac{d}{d_o}\right) \tag{2.3}$$

where η is called the decay factor and $PL(d_o)$ is the power lost at a reference distance d_o. The value of the decay factor depends on the frequency, antenna heights, and propagation environment. For example, in free space η equals to 2, and when obstructions are present, η is larger.

2.4.1.2 Log-normal shadowing

The path loss expression (equation 2.3) is an average value, and it does not consider the fact that the surrounding environmental clutter may be vastly different at two different locations having the same distance between transmitter and receiver. This leads to measured signals which are vastly different than the average value predicted by equation 2.3. Measurements have shown that at any value of d, the path loss $PL(d)$ at a particular location is random and distributed log-normally about the mean distance dependent value $\overline{PL}(d)$ [15, 30]. Thus, path loss $PL(d)$ can be expressed in terms of $\overline{PL}(d)$ plus $\mathcal{N}(0, \sigma)$, a zero-mean Gaussian distributed random variable (in dB) with standard deviation σ (also in dB) [126].

That is

$$PL(d)_{\ dB} = \overline{PL}(d) + \mathcal{N}(0,\sigma) = \overline{PL}(d_o) + 10\,\eta\log_{10}\left(\frac{d}{d_o}\right) + \mathcal{N}(0,\sigma) \qquad (2.4)$$

The log-normal distribution of path loss is due to the random *shadowing* effects which occur over a large number of measurement locations which have the same transmitter/receiver distance, but have different levels of clutter on the propagation path. This phenomenon is referred to as *log-normal shadowing*.

Finally, the received power $P_r(d)$ at distance d from transmitter is obtained with equation

$$P_r(d)_{\ dBm} = P_{t\ dBm} - PL(d)_{\ dB} \qquad (2.5)$$

where P_t is the transmission power.

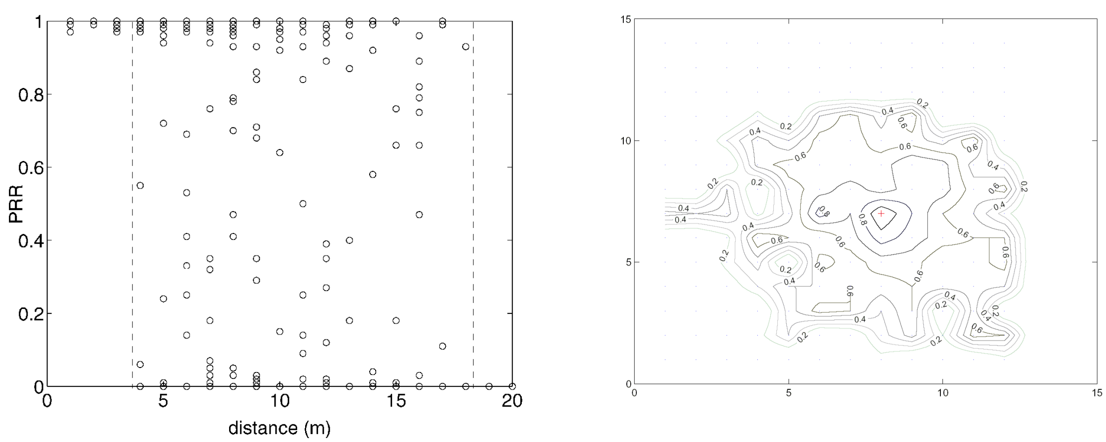
In practice, the values of η and σ are computed from measured data, using linear regression such that the difference between the measured and estimated path losses is minimised in a mean square error sense over a wide range of measurement locations and distances between transmitter and receiver. The value of $PL(d_o)$ (in equation 2.4) is obtained with close-in measurements or estimated with a free space assumption from the transmitter to d_o. The path loss model is used to study the transition region.

2.4.2 Transition region

Experimental studies have revealed the existence of three distinct reception regions in wireless link: connected (where all packets destined to a node can be received), disconnected (where none of packets destined to a node can be received), and transition (corresponding to all other cases). The nature of the transition region can have a major impact on the performance of upper-layer protocols and especially of the routing protocol. Also it needs to be considered for the simulations of WSN as several researchers [49, 142] have pointed out that the use of simple radio models may lead to wrong simulation results in upper layers. Below I describe what is at the origin of transition region and how it can be predicted in some scenarios.

2.4.2.1 Empirical study of transition region

Very often wireless transmission simulations and communication protocol operations in WSN are based on an assumption that the reception region can be represented by an ideal binary model *i.e.* a node in a given location can receive all packets destined to it or none of them [60, 85, 104]. However, the empirical studies have questioned the validity of this hypothesis.



(a) (b)

Source: Complex Behavior at Scale [49]

Figure 2.13: *Reception region characteristics. (a) Empirical measurement of PRR vs distance for multiple communicating pair nodes. (b) Contour of PRR from a central node.*

Figure 2.13 depicts the results of experiments [49, 96] which proved the existence of a so called transition region in which the packet reception rate (PRR) is quite erratic. It is important that upper layer protocols know the physical extent of this region because communication with nodes located in this region can lead to poor performance, for example, inefficient routing topologies [161].

2.4.2.2 Impact of transition region

The experimental and analytical studies revealed the impact of the transition region on the performance of communication protocols. First, asymmetric links[14] are predominantly found in the transition region [49]. The link asymmetry exacerbate the performance of a protocol which implements flooding and reverse path forwarding (*e.g.* Directed Diffusion, see 2.2.4.1). This is mainly because flooding relies on long hop links to propagate packets quickly through the network and those links are often asymmetric since they lie within the transition region. Hence, the reverse path forwarding is likely to fail.

Also, the link asymmetry may perturb the neighbour discovery process. In some WSN routing protocols (*e.g.* SPEED, see 2.2.4.2), upon reception of a beacon packet a node assumes that it can communicate with the source of that packet. However, if the sender lies in the receiver's transition region, then it is probable that although the node heard the beacon packet, the communication in opposite direction is not possible.

Another issue appears when nodes communicate over uncertain links. The protocol throughput can drop considerably after a few hops. If we consider a case that nodes send packets over link with 95% PRR, then overall packet throughput after 5 hops is 77%. In this case many ACK packets are required to ensure packet delivery.

Kotz *et al.* argue that the routing structures formed taking into account unreliable links can be very different from the structures formed based on a simplistic model [85]. Another problem depicted by Zhou *et al.* is that radio irregularity has a significant impact on routing protocols, but a relatively small impact on MAC protocols [161]. They found that location-based routing protocols, such as geographic routing perform worse in the presence of radio irregularity than on-demand protocols, such as AODV and DSR.

[14] An asymmetric link between two nodes occurs when one of them can transmit to the other but not vice versa.

Finally, discovering the transition region only on the basis of reception rate of control packets does not provide accurate results. It is mainly because PRR depends not only on the distance and terrain but also on the Error correction scheme used and generally, the smaller the packet, the less likely it is to encounter an error. Thus, neighbour probes using short packets do not accurately reflect the packet losses when using large data packets.

Hence there is a need for a realistic link layer model for WSNs which incorporates the effect of the transition region.

2.4.2.3 Model of transition region

The theoretical model of the transition region along with the reasons for its existence have been discussed by Zuniga [163]. The author claims that there are three factors contributing to the size and location of the transition region: the wireless channel, radio transmission characteristics, and the noise floor.

Details of this model along with the calculation which lead to the delimitation of the transition region are depicted in appendix B.

2.4.3 Time synchronisation in WSNs

The aim of the synchronisation protocol is to provide a common notion of time among the nodes. In this section the need for clock synchronisation is described together with the most popular synchronisation techniques proposed for WSNs.

2.4.3.1 Need for synchronisation in WSNs

The time synchronisation problem is a standard problem in distributed systems, but especially in WSNs. We can distinguish three main domains where synchronised clocks are required: at the interface between the sensor network and an external observer, among the nodes of the sensor network, and at the interface between the sensor network and the observed physical world.

In many applications, a sensor network interfaces to an external observer (a human operator or an autonomous computing system) for tasking, reporting results, and management. Tasking a sensor network often involves the specification of time windows of interest such as "only during the day". As a sensor network reports observation results to an external observer, the temporal properties of observed physical phenomena may be of interest. For example, the time-stamping of the samples gathered by sensors or times of occurrence of physical events are often crucial for the observer. Physical time is also necessary for determining properties such as speed or acceleration.

Time synchronisation is also required for intra-network coordination among different sensor nodes [92]. This includes security (*e.g.* authentication), data consistency (*e.g.* cache consistency, consistency of replicated data), concurrency control (*e.g.* atomicity, mutual exclusion such as TDMA) [61], and communication protocols (*e.g.* at-most-once message delivery). The energy efficiency of WSN can be improved by frequently switching sensor nodes or components thereof into power-saving sleep modes. In order to ensure seamless operation of the sensor network, temporal coordination of the sleep periods among sensor nodes may be required. This is called a *transceiver-receiver rendezvous problem* [13]. Also, many data-fusion[15] algorithms have to process sensor samples in order of their time of occurrence. However, sensor network may suffer from highly variable message delays and thus messages from distributed sensor nodes may often not arrive at a receiver in the order in which they were sent. These can be reordered according to the time of sensor readout only when nodes are synchronised. Methods for localisation of sensor nodes based on the measurement of time of flight or difference of arrival time of certain signals also require synchronised time.

In the third group of applications where node synchronisation is required we can distinguish data fusion which may extract higher-level information about

[15]Data fusion consists of the assembly of distributed observations into a coherent estimate of the original phenomenon.

an observed object (*e.g.* its size, speed, or shape) by correlating measurements from multiple locations. Also, sometimes one of the tasks of a WSN is to partition sensor samples into groups (that each represent a single physical phenomenon) when many instances of a physical phenomenon occur within a short time. Then the temporal relationships among sensor measurements are a key factor in performing this separation.

All these different requirements for time synchronisation have led to the development of many synchronisation protocols. The next section describes the main difficulties in designing such protocols and describes a representative synchronisation algorithm that I used as reference to estimate performance of my solutions described in chapter 3.

2.4.3.2 WSNs synchronisation difficulties and description of FTSP

Clock synchronisation algorithms face two problems: time-stamping jitter caused by delays in transmitting a packet and time errors due to the operating differences of hardware. The source of message time-stamping inaccuracy was analysed by Maroti *et al.* [97]. It includes sending, channel access, transmission, propagation, reception and receive time uncertainty. These errors are random and difficult to predict and eliminate especially in a network with many communicating nodes. Clock drifts may be compensated but it requires a periodic updating of the time information.

A popular synchronisation technique for WSNs is the Flooding Time Synchronisation Protocol (FTSP) described by Maroti *et al.* [97]. It utilises periodic flooding messages containing current local time information and originated at the elected coordinate which is the node with the lowest ID. Upon receiving this packet a node records the contained time-stamp and the time of arrival, and broadcasts the message to its neighbours after updating the time-stamp. Time-stamping is performed in the MAC layer to minimise delay variability and hence uncertainty. Each node collects eight (time-stamp, time of arrival) pairs and uses linear regression on these eight data points to compensate for frequency differences and offsets of the clock crystals. Results of experiments performed in an eight-by-eight grid of Berkeley Motes, proved the efficiency of FTSP. With this setup, the network synchronised in 10 minutes to an average (maximum) synchronisation error of $11.7\mu s (38\mu s)$, resulting in an average error of $1.7\mu s$ per hop. FTSP is robust to network failures, as it utilises the flooding of synchronisation messages to combat link and node failure. Moreover, FTSP can adapt to root node failure. However, FTSP may not be suitable for applications which do not require permanent clock synchronisation as the frequent flooding of temporal information requires significant energy.

2.4.4 Summary

In the early stage of WSNs many researchers applied solutions developed for ad hoc networks or adapted algorithms from other type of networks to implement routing protocols. However, these solutions turned out to be energy inefficient and thus many later routing protocols use approaches unique to WSN. In this chapter I presented the historical evolution of routing protocols prior to WSNs along with the most relevant examples of such protocols. I also included an overview of the challenges faced by routing protocols in WSNs, notably the constrained performance ability of the sensor nodes. These problems will be addressed in the rest of the thesis. I also reviewed the main routing protocols in WSNs and outlined the concept of cross-layer design.

Cross-layer technique is used to ensure energy efficiency of routing in WSNs where sensor nodes have limited computational and energy resources. Developing a feasible WSNs routing mechanism is still a topic of active research. Some researchers, including this author, believe that efficient WSNs routing strategies can best be implemented using a cross-layer design approach. Various existing implementations and architectures for cross-layer design have been described in this chapter. These topics will be considered in chapters 3 and 4. This chapter concluded with a discussion of the theoretical background to the wireless propagation model used in this research, transition region and the time synchronisation in WSNs. These topics are relevant to the cross-layer optimisation techniques described in later chapters. Chapter 3 concentrates on the mechanisms to incorporate time information into the network layer. Chapter 4 discusses techniques which use location information to enhance routing efficiency.

CHAPTER 3

Cross-layer routing incorporating time information

Time information is required by many protocols and applications in WSNs as discussed in section 2.4.3.1). There is thus a need to estimate it efficiently and to share it across the network stack. Because it is very rare for synchronisation not to be necessary in WSNs, I advocate integrating the synchronisation service into the routing layer. This chapter describes two novel routing protocols which enable energy efficient time synchronisation of nodes and use a cross-layer design approach:

- **CLEAR** – Cross Layer Efficient Architecture for Routing, which works well in cluster-based WSNs.
- **RISS** – Routing Integrated Synchronisation Service, intended for use in non-hierarchical WSNs.

I implemented and tested the energy efficiency and time precision of these solutions in different environments. Both protocols were deployed on sensor nodes using TinyOS with the aim of comparing their performance with other similar techniques. The protocols described in this chapter not only provide high precision of time synchronisation for low energy cost, but also achieve an excellent performance over a variety of network topologies.

3.1 CLEAR

The layered architecture may be unduly restrictive in the WSN context, where packets are routed by a cluster-based protocol. The autonomous operation of the network stack layers can lead to a significant drop of communication performance. For example I observed that the LEACH routing scheme [61] may cause a failure of the synchronisation protocol when it puts the node into low power mode. As a consequence, the WSN stops delivering measurements to the base station. Another issue that I identified was the high packet loss observed with the same routing scheme. This problem has two origins: the harshness of the communications environment and hardware limits of the nodes (which may be exacerbated by different configurations of LEACH). These problems can be encountered in a WSN when the routing is performed by a cluster-based protocol using duty-cycling of the nodes. I demonstrated these failures using the LEACH protocol which is one of the most widely used hierarchical routing schemes in WSNs. The analysis and solution to these problems is presented in this section.

3.1.1 Intra-cluster communication problem

Energy resources are the main constraint of a WSN. The power being drained from the nodes can be reduced by arranging sensor communication with a schedule, turning off the transmitter when data sending does not occur. This solution was implemented in many protocols, for example in LEACH where subordinate nodes communicate with cluster heads according to a TDMA scheme. However, in order to identify the allocated communication time slots, the sensor node clocks must be synchronised. The authors of LEACH propose to implement a separate synchronisation protocol and then share time information with the routing scheme. In my experiments this task was performed by the Flooding Time Synchronisation Protocol (FTSP) proposed by Maroti *et al.* [97]. The use of a synchronisation scheme independent of the routing protocol caused some of the problems I encountered with a deployed WSN. Also the energy efficiency of the routing protocol could be improved by using a different design approach.

I carried out all the described experiments with a 10 node network of sensor nodes uniformly distributed with a minimal inter-node distance of 5 m. I employed a commonly-used sensor network platform; namely the Tmote Sky sensor node [122], and the networking stack as implemented in TinyOS [64]. Its networking stack includes a default physical layer that supports single-bit error correction and double bit error detection capabilities. On top of this, its default MAC layer (LPL) implements a simple CSMA/CA scheme. The use of a contention based medium access protocol may seem inappropriate with the LEACH scheme because subordinate nodes communicate with cluster-heads according to TDMA schedule. Therefore most of the time nodes do not have to compete to access the channel. However, there are two main reasons why I decided to use this scheme. First, although the communication of nodes with the cluster-head is arranged in accordance with a TDMA schedule, collisions may still occur due to inter-cluster interference. Secondly, LEACH uses a CSMA/CA protocol in the setup phase. To save memory space I opt to re-use this scheme (and the associated code) as a solution to the inter-cluster contention problem during the communication phase.

For test purposes I implemented the LEACH protocol in the nesC programming language. The authors of LEACH only simulated the protocol and there was no available implementation of LEACH on real WSN nodes. LEACH requires network synchronisation and the authors of LEACH do not specify how it should be performed. Therefore I chose FTSP for this task because it is one of the most popular synchronisation protocols and also an implementation on TinyOS is provided by the authors. I also installed an application on the nodes which periodically collects humidity and temperature measurements from sensors integrated on the sensor nodes. This data is then routed with LEACH to the base station which is also a Tmote Sky sensor node. It differs from other nodes in the software installed. To test the routing scheme each node records the number of packets generated, whereas the base station stores the number of received messages. At the end of the experiment sensor nodes send data reports which are compared with the base station summary. These measurements are then examined for a quantitative study of the protocol performance.

First a series of tests demonstrated the data delivery efficiency of LEACH depending on its two parameters: the probability of becoming a cluster-head and the frequency of data packets. There may be many reasons for communication failure. First of all, interference can be caused by the signals sent by nodes from other clusters. Also the MAC layer does not prevent the hidden terminal problem. Finally, the computational resources of the nodes are too small to accept the high rate and long bursts of data. In consequence, if the receiver's processor is performing a task and the reception buffer or task queue is full, the message is discarded. This situation is unacceptable in applications where packet delivery must be guaranteed (*e.g.* intruder detection or medical applications).

Figure 3.1 shows how the cluster-head election process influences the packets delivery rate to the base station. In these experiments I collected packet failure delivery statistics as described above. I changed the probability of becoming a cluster-head for every test round of 6 hours. Sensors were taking measurements every 10 seconds and sending 3 packets per minute. The performance of LEACH drops with increasing probability of becoming a cluster-head. A higher density of cluster-heads increases the number of nodes trying to communicate with the base station. In consequence, the hidden terminal problem occurs more often and the base station may have problems handling many signals arriving at the one time.

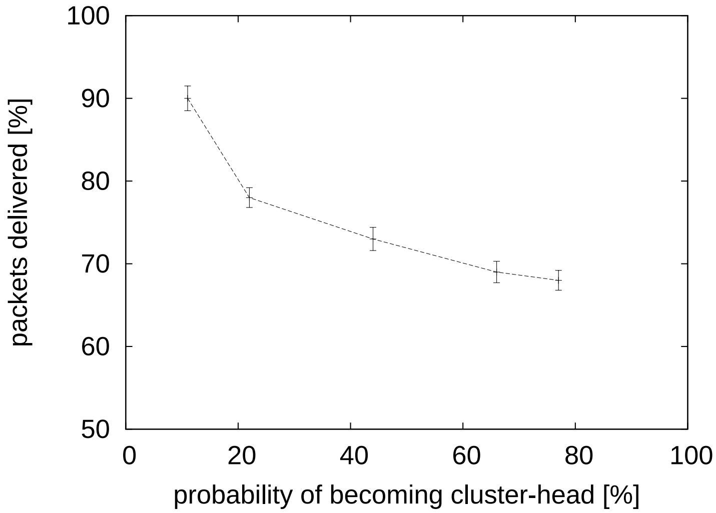


Figure 3.1: *Data delivery with LEACH protocol as a function of the probability of becoming cluster-head.*

I also observed that packet loss depends on the ratio of sensor measurement and data sending rates. Subordinate nodes can only communicate in the time slot assigned by the cluster-heads. In consequence, sensor measurements that cannot be sent have to be buffered until the next transmission. During the series of experiments described below, the cluster-head election probability was arbitrarily chosen to be 0.2 and packet delivery statistics are obtained by comparing the number of packets sent by every sensor node with the number of packets received by the base station. This time sensor nodes were sending packets with a constant rate of three packets per minute but for every experiment round of 6 hours I was varying the frequency of collecting sensor measurement. As can be seen in Figure 3.2, if more samples have to be stored before being sent then more packets are dropped before reaching the base station. This degradation of the overall performance can be explained as follows. The channel access protocols cannot sustain long bursts of messages, even if the total data rate is relatively low. I conclude that too long a sleeping period can have negative consequences on the network lifetime. Nodes turn off transceiver periodically in order to save energy. Then sensor samples are saved and sent in a longer data burst when the node wakes up. When the sleeping period increases, data delivery performance worsens. Lost packets have to be

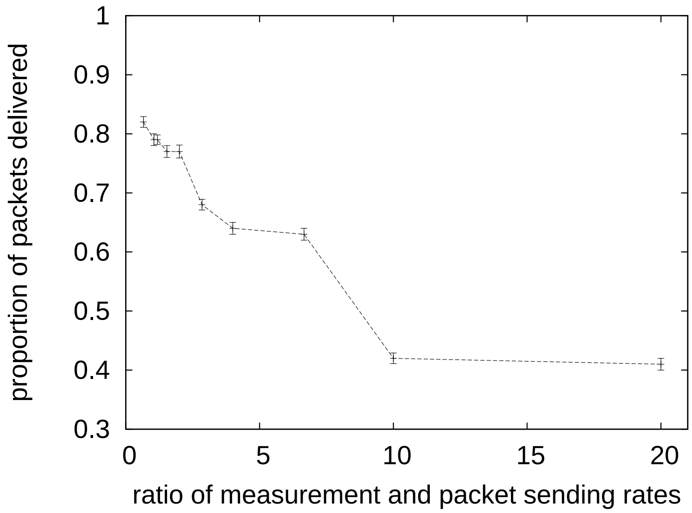


Figure 3.2: *Data delivery with LEACH protocol as a function of the packet sending rate.*

resent which results in additional energy drain from the sensor node. So some of the energy saved in a low power mode is consumed when lost packets are resent.

Another issue that I observed during experiments was the loss of synchronisation due to the duty cycle of the subordinate nodes. With FTSP, each sensor node sends periodic synchronisation messages and if LEACH initiates sleeping mode for subordinate sensor nodes, they cannot receive those packets and quickly become unsynchronised. That is why in my experiments with LEACH, low power mode was disabled. I address these two problems in designing the CLEAR protocol: failure of data delivery to the base station and loss of synchronisation of subordinate nodes. The solution is described in the next section.

3.1.2 A description of CLEAR

In my search I concentrated on the communication management scheme. I could solve the problems described in Section 3.1.1 with a traditional layered network solution. It would involve a packet acknowledgement mechanism to prevent data loss and a separate duty-cycle mechanism for FTSP to keep all of the nodes synchronised. This solution however has some drawbacks. First of all, it increases data traffic by adding ACK messages. Besides, independent sleep modes for routing and synchronisation schemes are energy inefficient. Also the duty-cycle mechanism for FTSP may require additional computations and can be complicated to implement.

The problems stated above led me to consider a solution that requires interaction of network stack layers. This idea motivated a variant of LEACH called CLEAR which combines both functions: routing and time synchronisation. The method I use to synchronise the clocks requires interaction with the MAC layer. As mentioned in section 2.4.3.1, many WSN applications can exploit knowledge of the global time of the distributed system (e.g. patient monitoring, object tracking, etc.).

The CLEAR protocol increases the energy efficiency of LEACH by optimising the duty cycle of the subordinate nodes in the clusters. It minimises the number of times subordinate nodes need to wake-up. The overall scheme operates as follows. Before starting the routing of messages with CLEAR, nodes need to synchronise their clocks. I used the Flooding Time Synchronisation Protocol (FTSP) for this purpose. It utilises periodic flooding messages originating at the elected coordinator which in my experiments was the base station. These contain timestamp information which is compared by the receiver with its local clock. With this method we can attain a high precision at discrete points in time. However, a very small difference in the clock frequencies can introduce errors for the global time estimation between synchronisation points. Thus FTSP implements linear regression to compensate for frequency differences of the clock crystals.

When node clocks are correlated the routing protocol elects cluster heads and other sensors join the closest cluster. Then the subordinate nodes receive the TDMA schedule from the cluster-heads and go to sleep until their communication slot arrives. However, the synchronisation information has still to be updated because of the clock drifts mentioned previously. My experiments showed that the combined operation of the synchronisation and routing protocols can extend the lifetime of the WSNs. Therefore CLEAR uses a modified scheme to align the crystals of subordinates nodes in the following way:

- A subordinate node transmits data to its cluster-head and waits a maximum of T seconds for an acknowledgement.
- Cluster-heads which receive packets from subordinate nodes check each packet's integrity and if there are no errors, stores them for retransmission to the base station. An ACK message is then sent which includes the synchronisation protocol information.
- After the arrival of the ACK message, the subordinate node updates its clock time on the basis of the information contained in the ACK packet. It then erases the original data and goes to sleep until the next communication slot.
- If the ACK packet is not received within T seconds, the node enters a low power mode but this time the data is stored and will be sent with subsequent sensor measurements at a future transmission time.

There are two reasons for integrating this acknowledgment scheme into LEACH. Primarily, this mechanism guarantees data delivery to the base station. My experiments carried out in a real environment, described in Section 3.1.1, showed me that LEACH may encounter a high rate of packet delivery failure. According to Zhao *et al.* [160] even reasonable link layer loss recovery is unable to mask the high data losses WSNs encounter in a real environment. Secondly, CLEAR makes the duty cycle of the subordinate nodes more efficient. The ACK packet integrates synchronisation data. It contains two fields: the source address and the global time of network. To increase energy efficiency, the cluster-head acknowledges all packets sent by a node within one time slot with a single message.

The CLEAR synchronisation method is based on the comparison of the global time value included in the ACK packet and the local clock value at the reception instance. Such a method may degrade precision because the medium access time is random. I implemented MAC layer time-stamping on the send and receive side in order to reduce message transmission jitter and in consequence the synchronisation error. The cluster-head performs time-stamping when a part of the ACK message was already transmitted whereas the subordinate node records the reception time of the global time field. With this method I attained a precision of 2 ms. The node which receives the ACK message updates the global time information, erases acknowledged measurements and goes to sleep. CLEAR minimises the number of times a node needs to awaken because it turns on the transceiver for both routing data and updating synchronisation information. In the next section I present an overview of CLEAR implementation.

3.1.3 Implementation of CLEAR

The CLEAR protocol can be implemented using any embedded operating system. The implementation presented here uses TinyOS technology [64], an open-source operating system designed for wireless embedded sensor networks which is described by Levis [88]. It features a component-based architecture which enables rapid innovation and reuse of already existing modules, such as sensor drivers, distributed services, data acquisition tools, and network protocols. Although TinyOS is open source and thus it is possible to modify the operating system internals, the protocol can be implemented without kernel modifications. To develop a protocol in TinyOS it is required to use some basic, standard components and add the new protocol modules. Thus, I refined the network stack to deploy some CLEAR functionalities and added custom protocol modules. The overview of TinyOS programming is given in the Programming Manual [88]. This section describes the implementation of CLEAR in TinyOS. Figure 3.3 shows the general diagram of CLEAR implementation which is described in more detail below. This figure shows only the components configured in the main CLEARAppC file and there are more subcomponents required for CLEAR which are not shown on the figure.

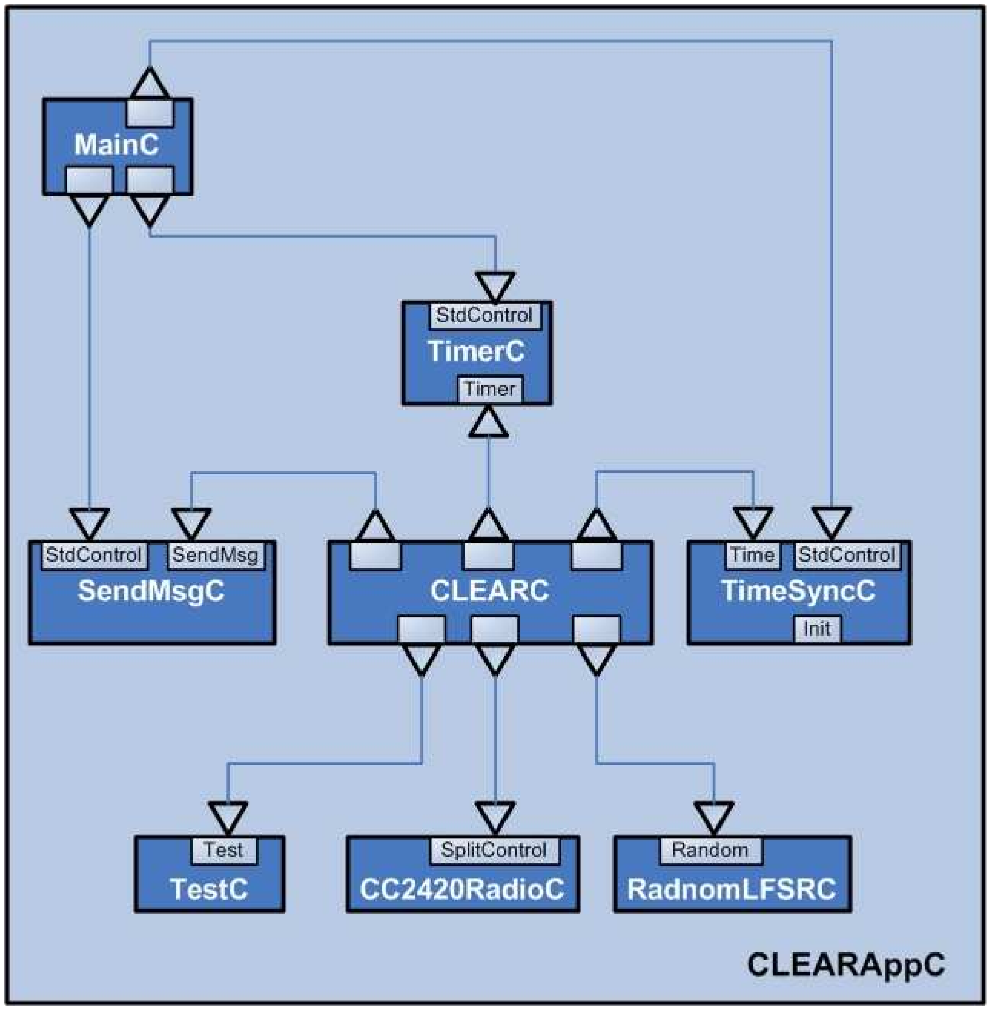


Figure 3.3: *Diagram of CLEAR protocol implementation - CLEARAppC configuration. Nodes represent components, and edges represent interface wiring. Triangles are labeled with the corresponding interface name.*

- **CLEARAppC** is the overall application configuration component. It connects the main functionality of the CLEAR protocol with other required modules.

- **CLEARC** is the core component of the CLEAR implementation. It manages CLEAR functionality, updates network topology, constructs and submits packets for sending, processes received packets, and handles time-stamping of outgoing packets.

- **MainC** is the component that is executed first in a TinyOS application. Its common StdControl interface is used to initialise and start TinyOS components.

- **TimerC** is the component that provides multiple instance of timers to the application. Theses are used by CLEAR to manage the TDMA schedule, communication phases, and the ACK message waiting time.
- **SendMsgC** provides multiple abstractions of message senders. It handles the sending of messages with different AM types.
- **TimeSyncC** is the configuration which manages time synchronisation tasks. It provides global time information to the CLEAR protocol.
- **TestC** is a component used for tests of CLEAR which gathers all required metrics.
- **RandomLFSRC** is an implementation of a 16-bit Feedback Shift Register [9] pseudo random number generator.
- **CC2420RadioC** is used for duty cycling and handles requests for stopping and restarting the CC2420 transceiver.

The setup phase and communication phase of CLEAR are implemented by the procedure shown in Listing 3.1 and 3.2 respectively.

Listing 3.1: *Setup phase of CLEAR protocol.*

```
startingSetupPhase = localTime + SETUP_PHASE_PERIOD
                               + DATA_COMM_PHASE_PERIOD
startingCommPhaseTime = localTime + SETUP_PHASE_PERIOD
leachRound++

threshold = EXPECTED_CLUSTER_HEAD/(NODES_IN_NETWORK
            - EXPECTED_CLUSTER_HEAD * (fmodf(leachRound,
```

```
            NODES_IN_NETWORK/EXPECTED_CLUSTER_HEAD)))
randNb = Random.rand()
IF randNb < threashold THEN//node is selected a cluster-head
    TimeSyncMode.setTimeSyncState|(STATE_CLEAR_CH)
    sendAnnouncementMsg()
    waitForCHSelectMsgs()
    calculateCDMASchedule()
    sendCDMASchedule()
    waitUntilNextSetupPhase(startingSetupPhase)
ELSE //node will not be cluster-head for the following round
    TimeSyncMode.setTimeSyncState|(STATE_CLEAR_NONCH)
    waitForAnnouncementMsgs()
    estimateClusterHeadID()
    sendCHChosenMsg()
    waitForCDMASchedule()
    waitUntilNextCommPhase(startingCommPhaseTime)
    waitUntilNextSetupPhase(startingSetupPhase)
ENDIF
```

Listing 3.2: *Communication phase of cluster-head using CLEAR protocol.*

```
    msgsToACK=0
    receivedDataMsg.receivedValues(dataPacket)


        bufferReceivedPacket(dataPacket)
        msgsToACK++
        wait(TRANSMISSION_SLOT)
        IF msgsToACK > 0 THEN
            sendACKMsg(msgsToACK,localTime)
        ENDIF
```

3.1.4 CLEAR performance

In this series of experiments I used the same WSN and data delivery performance measurement as described in section 3.1.1. In my tests comparing LEACH and CLEAR I was primarily interested in two aspects of wireless communication: energy consumption and packet loss.

The highest packet delivery ratio I could attain with real world experiments was 90% although it could drop to as low as 40%. Therefore some mechanism of assured delivery is required and CLEAR integrates into LEACH a message acknowledgement mechanism. Thus at the price of increased channel occupancy I guarantee packet delivery to the base station. Even if some messages are lost with CLEAR, the acknowledgement mechanism ensures the delivery of sensor measurements to the base station.

I also analysed the influence of CLEAR on the network lifetime. As mentioned already, CLEAR makes the duty cycle of the subordinate nodes more efficient, and so the network can operate for longer periods than when using LEACH with the same initial energy resources. In order to predict system longevity, I measured the current consumption of the Tmote Sky sensor nodes. To do this I used the voltage measurement circuit described in the appendix A.

First, the sensor nodes were operated using the LEACH and FTSP protocols. I observed that sensor nodes became quickly unsynchronised when LEACH initiated the periodic low power mode due to FTSP messages not being received. Hence I could not run experiments with LEACH and low power mode for a long time. Instead, the protocols are implemented with a layered architecture and the sensor nodes are not put to sleep. On average, the nodes drained a current of 20 mA each, so with two typical AA batteries (2000 mAh) as a power source, network longevity can be estimated to be 100 hours.

In contrast, CLEAR facilitates the nodes entering into low power mode. Then the current consumption drops to about 250 μA. So with nodes sending packets every two minutes, CLEAR can extend the network lifetime for the same battery resources up to around 840 hours (when the cluster-head election probability equals 10%). This value depends on two parameters of the routing scheme: the fraction of nodes being made cluster-heads and the frequency of packet transmission. Figure 3.4 shows the variation of WSN lifetime as a function of these variables. Diminution of the cluster-head election probability has a positive influ-

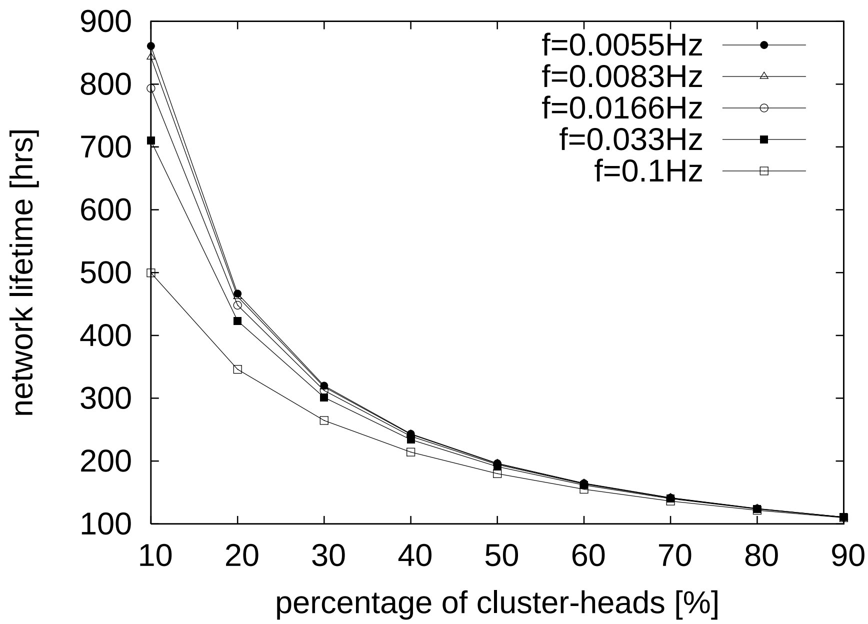


Figure 3.4: *Network lifetime with CLEAR protocol using different packets transmission rate f.*

ence on the network longevity. With more nodes operating in low power mode, the energy draining from the system drops. Also, frequent data transmission by subordinate nodes can limit the lifetime of the WSN.

High packet loss was encountered with the LEACH protocol when many packets had to be stored in a buffer before being sent because of the long sleeping period. It may be a critical problem for the applications requiring high measurement frequencies. CLEAR makes the duty cycle more energy efficient because nodes send packets and are synchronised during the same wake up interval. Besides it ensures data delivery to the base station so the nodes can be put into sleep for a longer time without decreasing the measurement frequency.

3.1.5 CLEAR: summary

The experiments performed in a real environment showed me that the traditional layered architecture may make the routing protocol inefficient in terms of energy consumption. I discovered also that LEACH, a hierarchical routing protocol, may encounter high packet loss. Thus I proposed to integrate synchronisation, routing, MAC time-stamping and an acknowledgment scheme into a new, cross-layer designed protocol called CLEAR. It drains less energy from the nodes compared with LEACH and in consequence extends the network lifetime. It also guarantees data delivery to the base station.

In clustered networks some nodes become cluster-heads and they are responsible for managing neighbours. The other nodes join clusters and perform tasks according to the commands sent by their cluster-heads (*e.g.* they send samples in accordance with the schedule generated by the cluster-head). Thus the hierarchy importance increases in the downstream direction because a node is responsible for managing its upstream neighbors. Also a clustered architecture requires that nodes communicate over bidirectional links *i.e.* data must be sent over the same links in both directions in these networks. For example the cluster-heads transmit the TDMA schedules and data requests in the upstream direction but the subordinate nodes send sensor samples over the same links in the downstream direction. For these reasons CLEAR is best suited to such networks, since it also requires bidirectional links. With CLEAR, a cluster-head sends the synchronisation information in the reverse direction over the same link as was used by a subordinate node to send sensor samples.

3.2 RISS

The time synchronisation problem needs to be considered in a distributed system, but especially in WSNs where a common time reference is necessary for duty cycling of nodes and where most applications require time stamping of the samples gathered by sensors. Additionally in WSNs this issue must be solved with limited computational, communication and energy resources. There are many protocols in the literature that address this problem [79, 87, 97]. Most of them comply with the traditional layered architecture using a protocol-specific sequence of packets and a strict network stack hierarchy. These solutions are efficient in terms of synchronisation precision but in WSNs we have also to consider the power cost of a protocol. Compared to these protocols, the network synchronisation can be obtained in a more energy efficient way when the synchronisation service is integrated in the routing layer. This design approach is also easier to incorporate in new applications.

Hence, time synchronisation protocols which use cross-layer information sharing have been proposed [65, 121]. In those scenarios, the advantage of granularity of the network stack is still retained. However, time information estimated by one layer can be shared with other layers. A novel method employing such an approach and called RISS (Routing Integrated Synchronisation Service) is described below. This protocol integrates the synchronisation service into the routing layer to achieve three goals:

- to minimise the overhead of the protocol
- to minimise the awake time of the transceiver and in consequence its energy dissipation
- to achieve good time synchronisation
- to achieve robust protocol performance across a range of values for the duty cycle of the nodes.

3.2.1 An outline of RISS

The main purpose of WSNs is to interact with the environment by sensing or controlling physical parameters [79]. In most application scenarios (*e.g.* monitoring of the outdoor environment, habitat, offices, systems, buildings, and industrial sites) nodes perform the computational and transmission tasks periodically. It is rarely the case that the node stays active while it waits for an event to happen but even in those cases it must periodically broadcast a beacon message to maintain network connectivity. This periodic transmission being a common property of WSNs applications, I propose to exploit it to obtain an overall network time reference. Less overhead is required to estimate neighbour's clock frequency than to generate a global network time. Accordingly, since the previously mentioned monitoring applications do not require overall time reference, synchronisation supported by RISS is limited to the estimation of the neighbours' clock frequency.

Typically, data is forwarded regularly to the *downstream* nodes (in the direction towards the base station) from the *upstream* nodes (in the reverse direction). However with RISS a sensor node synchronises to the upstream neighbour. I chose this approach because of the following reasons. First, it reduces the protocol overhead because the downstream node can estimate the frequency of a neighbour's clock on the basis of periodic packets sent by the neighbour. So RISS does not require any additional packets and in consequence the energy cost of the protocol is reduced. Second, in WSNs data is mostly transmitted in downstream direction and it is possible that a pair of nodes can communicate directly only in that direction as they are connected by an asymmetric link (see section 2.4.2.2). Thus the synchronisation error between these two nodes is minimised if they synchronise on the basis of the synchronisation data sent over direct link in the downstream direction. Finally, it may even happen that there is no multi-hop path from the downstream to the upstream node even thou these nodes can communicate directly in the opposite direction. In this situation, the synchronisation information may only be sent in the downstream direction.

So in order to synchronise, a node maintains a table with the clock information of every upstream neighbour. To construct that table, the downstream node uses the periodicity of the operation of the upstream neighbour and the time information added to the packet by the sender. So the task of the upstream node is to add information to the message that would facilitate the time alignment of nodes. However, I recommend adding not a time-stamp of the sending instant, but rather the time which has elapsed since the last local periodic interrupt clock (the reasons for that are described in section 3.2.2.1). Then the receiver after collecting multiple messages can calculate the neighbour clock frequency with respect to its local clock. Below, a more detailed description of RISS is provided.

3.2.2 Detailed description of RISS

Variable	Meaning
T_i	Wake-up instance of the transceiver to execute a periodic task
P	Duty-cyle period of the transceiver
W_i	Time elapsed between wake-up of the transceiver and sending of the packet
R_i	Packet reception instance
Q	The number of past packets processed
$F_{r \to t}$	Frequency of the transmitter clock
$I(k)$	Time occurrence of an observed event expressed at the node k

Table 3.1: *Variables used in the mathematical formulas.*

As mentioned previously the downstream node estimates the frequency of its upstream neighbours. So, for example the base station has to evaluate the clocks of the nodes which report data directly to it. The task of the upstream sensor nodes is to perform some periodic operation (*e.g.* to awaken the transceiver) at time T_i and to send a packet (Figure 3.5). The task period P is known to the downstream receiver and it may be decided before the deployment of the network or changed operationally and reported to all nodes. In this case the value travels in the upstream direction so before reaching a node it is learned by the downstream neighbour.

It takes an amount of time W_i for the sender to actually transmit the packet. This waiting time is inserted into the message just after the transmission of the SFD[1]. When the receiver captures the SFD of the packet it reads the local clock

[1]Start of Frame Delimiter (SFD) is the 8-bit value (0xA7) marking the end of the preamble of

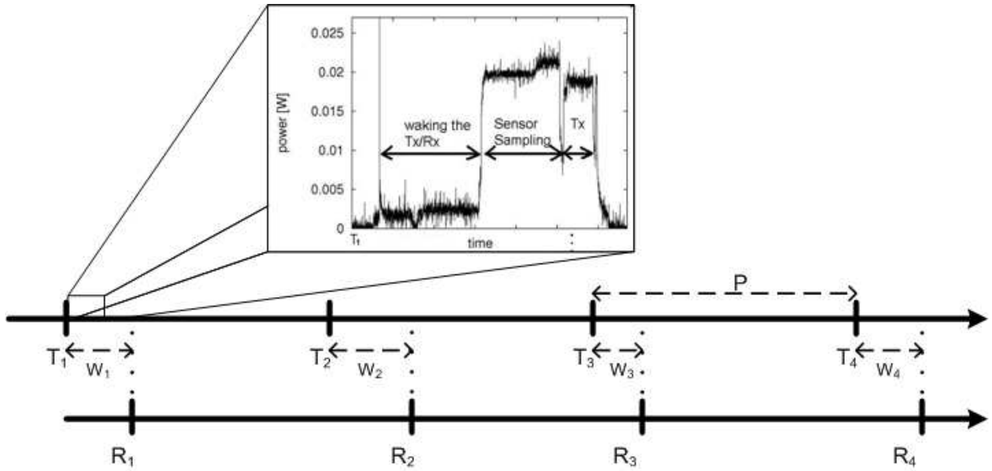


Figure 3.5: *Time line of the operation of the receiver (bottom) and transmitter (top). Every P seconds the sender wakes up the transceiver at its local time T_i, samples the sensor, and transmits the SFD of the packet at local time $T_i + W_i$. Sender adds the value W_i to the message and turns off the transceiver. The receiver hears the SFD at the local time R_i.*

and saves this value (R_i) for further estimation of the sender's clock frequency. A downstream node needs to collect several packets from a single upstream neighbour in order to synchronise with its clock.

The repetitive operation of the protocol overcomes the problem of hardware clock errors. The aim of the time field included in the packet is to minimise the synchronisation error due to time-stamp jitter.

3.2.2.1 Minimisation of time-jitter error

Time-jitter error has multiple origins. These include the time taken to assemble a packet and to submit it to the MAC layer which is unpredictable, and depends on the CPU usage. Also the channel access time is random whereas the propagation time [2] is highly deterministic in WSNs and increases with the distance between the two nodes. To synchronise the clocks I propose to measure those random quantities (the values of W_i in Figure 3.5) and send them in a packet to a receiver which will estimate the frequency of the sender clock. Sending information just about the random time delay (W_i) and not the clock reading at the

an IEEE 802.15.4 frame.

[2]Propagation time is the time it takes for the message to transmit from sender to receiver once it has left the sender

time of transmission ($T_i + W_i$) has two advantages. Firstly, this number occupies a small range in comparison to the possible clock scope. Thus, sending it requires less energy and packet space. In my experiments I observed that the maximum nondeterministic delay of the transmission was 566 ticks using a 32kHz clock. We need 10 bits to transmit this number instead of the 32 bits required for sending the time-stamp. This is important in WSNs where most of the energy resources are consumed by the transceiver [46]. Also in WSNs, the packet space is very limited. For example in Tmote's implementation of the network stack the user has 28 bytes for application data. So saving even 22 bits of the header might have a substantial benefit. In the next paragraph I describe how the receiver synchronises with its neighbours.

3.2.2.2 Receiver operation

The downstream node records the information about the last Q packets received from a sensor node in order to estimate its clock frequency. This data is stored in a $Q \cdot l$ array where l is the number of upstream neighbours. For every packet received, the node must save the corresponding W_i and R_i values. After that, the sensor node estimates the clock frequency of the most recent transmitter from the set of W_i and R_i values stored in the table using the method below. This calculation is performed every time a packet is received. I compared two different techniques to estimate the sender's clock frequency.

Linear regression: I implemented linear regression using a standard optimised algorithm [124]. So, if I want to estimate the frequency of the transmitter clock $F_{r \to t}$ with the Q most recently received packets, the sum sy of receiver time-stamps of the packets reception is given by:

$$sy = \sum_{i=c}^{c-Q+1} R_i \tag{3.1}$$

Every time-stamp R_i corresponds to the time of transmission of the SFD expressed using the sender's clock. If i is the packet sequence number, the sum of the SFD transmission time-stamps can be expressed as follows:

$$sx = \sum_{i=c}^{c-Q+1} i \cdot P + W_i \tag{3.2}$$

So the frequency of the transmitter clock $F_{r \to t}$ can be obtained with the following equation:

$$F_{r \to t} = \frac{\sum_{i=c}^{c-Q+1} t_i \cdot R_i}{s} \tag{3.3}$$

where $t_i = i \cdot P + W_i - \frac{sx}{Q}$ and $s = \sum_{i=c}^{c-Q+1} t_i^2$

These calculations turned out to be very resource demanding. In particular the division by large integers in equation 3.3 requires a considerable amount of time. Because of the time expenditure and in consequence, energy inefficiency of this method I propose another technique for estimating the frequency of the sender's clock for use in the RISS protocol.

Fast approximation: I call the frequency of the transmitter clock expressed at the receiver $F_{r \to t}$. For every packet sent with an inter-packet period equal to P we can write:

$$R_i - R_{i-1} = F_{r \to t} \cdot (P - W_{i-1} + W_i) \tag{3.4}$$

or

$$F_{r \to t} = \frac{R_i - R_{i-1}}{P + (W_i - W_{i-1})} \tag{3.5}$$

I want to avoid the division operation in equation 3.5 for computational efficiency. For convenience I write $R_i - R_{i-1} = \Delta R_i$ and $W_i - W_{i-1} = \Delta W_i$. Hence, equation 3.5 becomes:

$$
\begin{aligned}
F_{r\to t} &= \frac{\Delta R_i}{P + \Delta W_i} && (3.6)\\
&= \frac{\Delta R_i}{P} \cdot \frac{P}{P + \Delta W_i} && (3.7)\\
&= \frac{\Delta R_i}{P} \cdot \frac{1}{1 - (-\Delta W_i / P)} && (3.8)\\
&= \frac{\Delta R_i}{P} \cdot \left(1 + \left(\frac{-\Delta W_i}{P}\right) + \left(\frac{-\Delta W_i}{P}\right)^2 + \left(\frac{-\Delta W_i}{P}\right)^3 + \ldots\right) && (3.9)
\end{aligned}
$$

which converges provided $|\Delta W_i| < P$. In this context[3], $|\Delta W_i| << P$ and so $F_{r\to t}$ is well approximated by $\hat{F}_{r\to t}$ where

$$
\begin{aligned}
\hat{F}_{r\to t} &= \frac{\Delta R_i}{P} \cdot \left(1 - \frac{\Delta W_i}{P}\right) && (3.10)\\
&= \frac{\Delta R_i}{P} - \frac{\Delta R_i \Delta W_i}{P^2} && (3.11)
\end{aligned}
$$

The second term is well approximated as $\Delta W_i / P$ since $\Delta R_i \simeq P$. Hence:

$$\hat{F}_{r\to t} \simeq \frac{(R_i - R_{i-1}) - (W_i - W_{i-1})}{P} \qquad (3.12)$$

I calculate only the numerator of that expression. Defining $F'_{r\to t} = P \cdot \hat{F}_{r\to t}$ and averaging that value over the Q most recently received packets I obtain:

$$\overline{F'_{r\to t}} = \frac{\sum_{i=c}^{c-Q+1} [R_i - R_{i-1} - W_i + W_{i-1}]}{Q} \qquad (3.13)$$

I show in sections 3.2.3 and 3.2.4 how the obtained value of $\overline{F'_{r\to t}}$ can be used for duty cycling of nodes and for estimation of a common time reference. Before-

[3]I determined experimentally the maximal value of ΔW_i to be 237 whereas minimum value of P is 32768 when duty cycling period equals 1s.

hand, in the next paragraph I describe how the novel synchronisation service can be integrated with the routing protocol.

3.2.2.3 Integration of RISS into network layer

Incorporating the synchronisation service in the routing protocol at the network layer results in an efficient implementation since the network architecture is managed by this layer. In the synchronisation method I propose, every node must first discover its upstream and downstream neighbours. This information can be deduced from the routing tables. I propose that if the duty cycling causes the loss of a packet the receiver should not go to sleep until the arrival of the next message (see section 3.2.3). This event can be discovered via a comparison of the sequence numbers of received packets. This value is included in the packet header by many WSN routing protocols so we can use it for the duty cycling control.

3.2.3 Duty cycling of the node

Duty cycling is one of the most common techniques used in WSN to save the energy resources of the sensor nodes [12, 79, 155]. As stated previously, the transceiver drains more energy than other subsystems from the sensor node. For this reason it seems reasonable to turn it off periodically. However, in order to maintain the connectivity of the network we must ensure that when the transmitter wants to forward a packet, its destination is awake. There is thus a tradeoff between keeping nodes asleep for the maximum possible time and minimising the packet loss due to the receiver not listening. Thus it is necessary that the communicating sensor nodes have the same time reference and wake up at the same instance to communicate. We can use the frequency of the sender's clock as estimated with the RISS protocol to synchronise the duty cycling of the nodes in the following way.

When the downstream node captures a sufficient number of packets from its upstream neighbour to estimate its clock frequency, it calculates the arriving time of the next packet and goes to sleep. This estimation can be done as follows. After reception of packet i, we can expect that the arrival time-stamp of packet $i+1$ is:

$$R_{i+1} = R_i + P \cdot F_{r \to t} + F_{r \to t} \cdot (W_{i+1} - W_i) \tag{3.14}$$

However, we cannot predict the term W_{i+1} which is stochastic. Thus I propose to replace the difference $(W_{i+1} - W_i)$ by a minimum of that value over the last Q packets $\min_{a\epsilon(0:Q-1)}(W_{i-a} - W_{i-a-1})$. I do that to minimise the packet loss due to the transceiver waking up after the packet's arrival. In order to reduce the error of calculation of the neighbour's clock frequency I propose to combine the results of multiple estimates. Also, the term $\min_{a\epsilon(0:Q-1)}(W_{i-a} - W_{i-1-a})$ corresponds to a small value so we can neglect its multiplication by $F_{r \to t}$ which is very close to one. So equation 3.14 becomes:

$$\hat{R}_{i+1} = R_i + P \cdot F_{r \to t} + \min_{a\epsilon(0:Q-1)}(W_{i-a} - W_{i-1-a}) \tag{3.15}$$

The arrival of the next packet can be predicted with equation 3.15. However, a packet loss may still occur because the value $\hat{R}_{i+1}$ is estimated on the basis of the arrival times of the most recent Q packets. Thus, if for example the channel occupancy drops the next packet will have a smaller channel access time than the previous Q packets. Thus, it will be transmitted before estimated time $\hat{R}_{i+1}$ and not heard by the still sleeping receiver. To prevent such packet loss I propose to awake the receiver earlier than the estimated time $\hat{R}_{i+1}$ by a guard interval G which I determine empirically. For a general application scenario it would be most reasonable to make the value of G adaptive and either to increase it when the reception of a packet is missed or decrease its value when a receiver spends a long time idly listening. However, RISS can be used in networks with asymmetric links where the receiver cannot acknowledge the reception of a packet over the same link. If G were adaptive, losses of several consecutive packets could occur. In consequence, if the link is asymmetric so that the receiver cannot directly inform the sender about the reception failure, a significant amount of data can be lost. Therefore I propose to use a fixed value of G and to keep the receiver awake after a reception failure until the arrival of the next packet. With this mechanism two successive packet losses cannot occur.

The equation 3.15 applies to the situation when the linear regression is used for the neighbour clock frequency (section 3.2.2.2). For the case of fast approximation (section 3.2.2.2), the arrival time of next packet is calculated differently. For this purpose, equation 3.13 can be incorporated into equation 3.15 in order to reduce processing time. Also, we can replace the term $P \cdot F_{r \to t}$ with the value of $\overline{F'_{r \to t}}$ which will save further time by avoiding division and multiplication by P. So finally, the estimated time of the next arrival of packet can be approximated efficiently as:

$$\hat{R}_{i+1} = \hat{R}_i + \overline{F'_{r \to t}} + \min_{a \epsilon (0:Q-1)} (W_{i-a} - W_{i-1-a}) - G \tag{3.16}$$

where $\overline{F'_{r \to t}}$ is obtained with equation 3.13.

$\hat{R}_{i+1}$ is the state variable denoting the required wake up time of the node. It is updated after every reception of a packet. When a receiver has multiple upstream neighbours, it must keep a record of estimated, future arrivals from all of them. Whenever it receives a packet, it predicts the arrival time of the next message from the same sender and goes to sleep until the projected time $\hat{R}_{i+1}$ of the next message to arrive from all upstream neighbours.

3.2.4 Estimation of event time correlation

Another major purpose of obtaining a common network time reference in WSN is the ability to time-stamp events. In most WSNs applications the user must know the time of occurrence of a sensed event. I propose a method of generating time-stamps which requires only a little protocol overhead. In section 3.2.2.2 I describe how every node can estimate the frequency of its upstream neighbours with RISS. This information may be used at each node to estimate the time-stamp, expressed in local clock units, of the received samples from an upstream neighbour. To obtain that value the receiver needs to know the time, expressed with the clock units of the upstream neighbour, elapsed between the occurrence of the sample and the instance of sending the packet containing the sample. Then the receiver is able to express that period in its local clock units and by consequence precisely estimate the instance of acquisition of the data sample. To obtain that value it must substract the calculated value of elapsed time from the instance of reception of the SFD of the packet. Thus I propose to insert into the packet a field which is updated by every node on the communication chain between the source of the packet and the base station to reflect its own local clock. Initially, the sensor node k records the time $E(k)$ elapsed between an observed event (which can be sampling of the sensor, a packet reception *etc.*) and the periodical clock interrupt ($T_i(k)$) described in section 3.2.2 in that field. Then the receiver (node $k+1$) computes in its local clock units the period between the event observed by the sender and the SFD of the received packet at time $R_i(k+1)$. This value can be obtained as follows. If the time value sent by the node k is $E(k)$ and the frequency of the sender clock estimated at the receiver (see section 3.2.2.2) is $F_{k+1\to k}$, then using the linear regression method (section 3.2.2.2) the time of occurrence $I(k+1)$ of the observed event can be obtained with the following equation:

$$I(k+1) = R_i(k+1) - F_{k+1\to k} \cdot [W_i(k) + E(k)] \tag{3.17}$$

and using fast approximation (section 3.2.2.2) the estimate is:

$$I(k+1) = R_i(k+1) - \frac{\overline{F'_{k+1\to k}} \cdot [W_i(k) + E(k)]}{P} \tag{3.18}$$

where $W_i(k)$ is the value sent in the packet (see section 3.2.2.1).

If the receiver of the packet is not the base station, it forwards the value $E(k+1)$ to the downstream neighbour (node $k+2$). The value $E(k+1)$ corresponds to the time elapsed between the initial sensor sampling, expressed at node $k+1$, and transmission of the packet and is given by:

$$E(k+1) = F_{k+1 \to k} \cdot (W_i(k+1) + E(k)) + P \tag{3.19}$$

Then the receiver (node $k+2$) will collect the SFD of the packet and repeat the update of the time field before the transmission. So the time of occurrence $I(k+2)$ of the observed event expressed at node $k+2$ can be obtained with the following equation:

$$I(k+2) = R_i(k+2) - F_{k+2 \to k+1} \cdot [W_i(k+1) + E(k+1)] \tag{3.20}$$

and using a fast approximation (section 3.2.2.2) the estimate is:

$$I(k+2) = R_i(k+2) - \frac{\overline{F'_{k+2 \to k+1}} \cdot [W_i(k+1) + E(k+1)]}{P} \tag{3.21}$$

where $R_i(k+2)$ is the reception time of the packet from node $k+1$ containing the value $E(k+1)$ and $W_i(k+1)$ is the random delay of that packet. This operation continues until the message reaches the base station.

3.2.5 Implementation details

The RISS protocol can be implemented using any embedded operating system. The implementation presented here uses the TinyOS technology [64] described by Levis [88]. I developed two applications to demonstrate the performance of RISS. One shows the use of RISS for duty cycling of the nodes and the second enables the time-stamping of events. The experimental scenarios used to test these protocols are described in section 3.2.6.4. A general description of these implementations is given below.

3.2.5.1 Implementation of RISS for node duty cycling

To facilitate deployment of the code on sensor nodes, I decided to integrate the functionalities of every node into a common program, installed on each sensor node. A node then becomes the base station or a subordinate sensor node based on its unique ID value which is assigned at the code installation. Figure 3.6 shows the components of RISS and the interfaces by which they are wired. Below I provide an overview of these modules:

- **RISSDutyCyclingAppC** is the overall application configuration component. It connects the main capabilities of the RISS protocol with other required modules.

- **RISSDutyCyclingC** collects sensor samples, manages duty cycling of the transceiver, constructs and submits packets for sending, handles time-stamping of outgoing packets, and processes received packets. It is also responsible for handling user button events and the sending of test results.

- **RISSScheduleC** is the module that performs RISS time processing. It estimates frequencies of neighbours' clocks, predicts the arrival time of the next packet and decides when the node can be put to sleep.

- **MainC** is the configuration that wires[4] the boot sequence implementation to the scheduler and hardware resources. It exports the *Boot* and *SoftwareInit* interfaces for application to wire to.

- **AMReceiverC** is a generic configuration which handles reception of packets of predefined protocol types[5]. It signals whenever the packet layer receives

[4]In TinyOS the application components are connected(or wired) through the defined interfaces.

[5]The receiver can identify the packets generated with a given protocol by comparing the *Active Message* field of the packet which is a protocol identifier used in TinyOS.

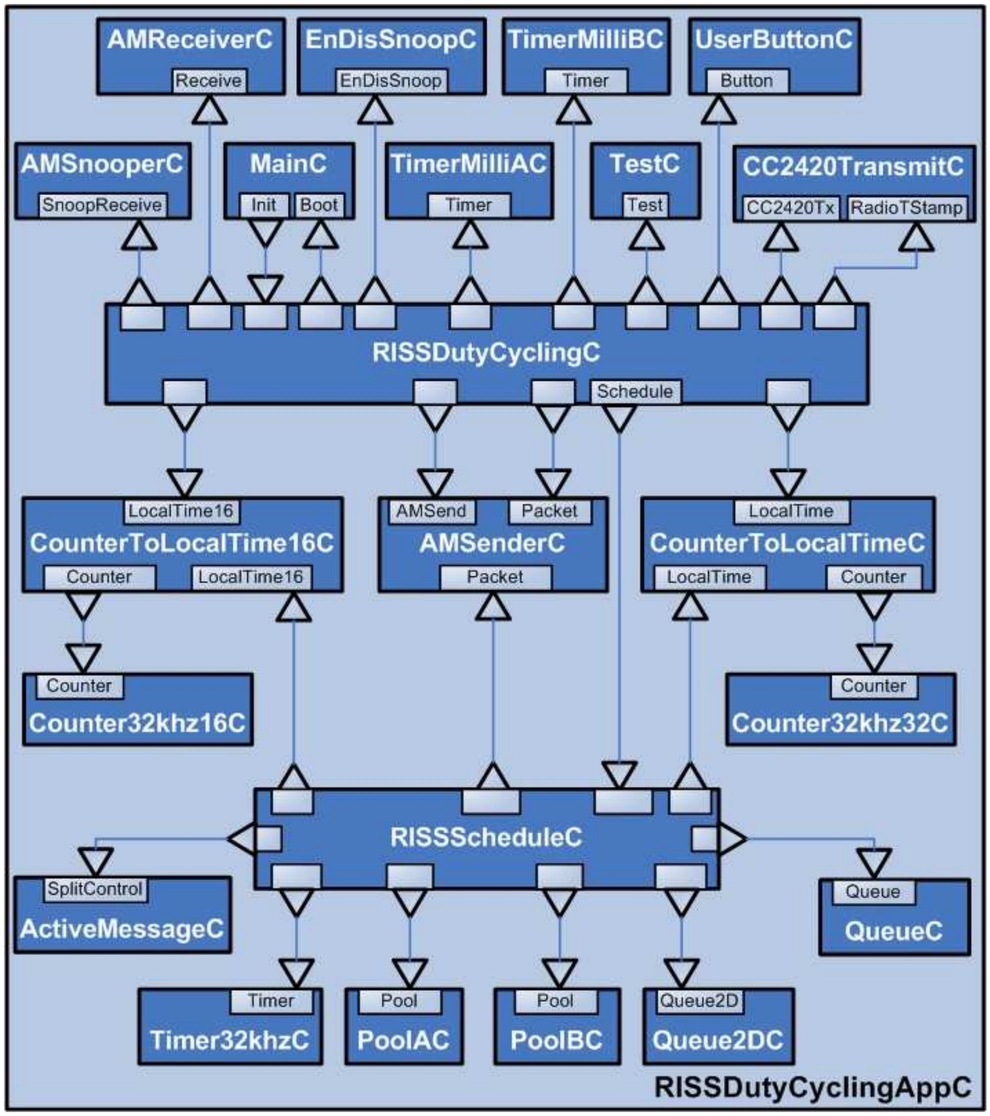


Figure 3.6: *Diagram of RISS implementation for duty cycling - RISSDutyCyclingAppC configuration. Nodes represent components, and edges represent interface wiring. Triangles are labeled with the corresponding interface name.*

an active message of the corresponding AM type whose destination address is either the local address or the broadcast address.

- **AMSnooperC** is similar to the AMReceiverC component. The only difference is that it signals reception of a packet whose destination address is different than the local address. This module is needed to reduce channel congestion and interruption of RISS processing by an arrival packet. A node, prior to sending a packet, listens to the channel to avoid simultaneous transmission with other sensor nodes.

- **EnableDisableSnoopC** is used to enable listening of the channel by subordinate nodes.

- **TimerMilliAC and TimerMilliBC** are both TimerMilliC generic configurations[6] provided by TinyOS. TimerMilliC is the virtualised millisecond timer abstraction. TimerMilliAC is used to manage periodic operation of the node (see section 3.2.2). When it signals an event the node initiates the cyclical task. The TimerMilliBC timer provides time information at the initial setup of the experiment when the node listens to the channel to minimise traffic congestion.

- **TestC** is a component which handles test results. At the end of an experiment it processes predefined metrics and submits them for transmission to the user.

- **UserButtonC** provides a button interface for the user button on Moteiv's Tmote Sky sensor nodes and as a consequence it handles the button events. When the experiment is finished the user can press the button and the results will be sent on the channel.

[6]The TinyOS concept of generic configuration is described in TinyOS programming manual [88]

- **CC2420TransmitC** is the implementation of the transmit path for the Chipcon CC2420 radio, which is the transceiver used for the experiments. I connect this component to RISSEventTimeStampingC with the goal of providing the option of time-stamping the packet after its SFD was sent.

- **CounterToLocalTime16C and CounterToLocalTimeC** convert a 32-bit LocalTime to a Counter and they differ by the LocalTime range which is respectively 16-bit and 32-bit. They export the *LocalTime* interface for the application which provides the local time information to the node as a 16-bit or 32-bit value.

- **Counter32khz16C and Counter32khz32C** provide at 16-bit or 32-bit respectively counter at 32768 ticks per second.

- **ActiveMessageC** is just a configuration that renames a particular radio chip's active message layer, in this case CC2420ActiveMessageC. It provides interfaces to handle radio communication. The configuration also enables the AM layer to multiplex access to the radio.

- **Timer32khzC** provides a 32-bit, 32kHz timer for the application. When the timer generates an interrupt, the RISS schedule wakes up the transceiver because the node expects a packet arrival. Also after reception of a packet, if the node is put to sleep the timer is initialised with the expected time of arrival of the next packet.

- **PoolAC and PoolBC** are both PoolC generic configurations. PoolC is a general dynamic memory pool component. PoolAC is used to store scheduler information necessary to calculate the time of arrival of the next packet from every neighbour. PoolBC provides memory for recording the last Q packets from every neighbour (see section 3.2.3). They need to be stored for use in the estimation of the time of arrival of the next packet.

- **QueueC and Queue2DC** are both general FIFO queue components, whose queues have bounded sizes. Together with PoolC components they provide mechanisms for storing the scheduler's data and the last Q packets heard by a node respectively. PoolC provides memory space for the scheduler's data (or for an arriving packet) whereas QueueC (or Queue2DC) stores a pointer to this memory slot at the end of the queue. So the order of arrived packets (or scheduler's data) can be obtained from the QueueC (or Queue2DC) component. The only difference between the QueueC and Queue2DC components is that the former represents a single queue whereas Queue2DC provides many queues, one for each neighbour.

The estimation of the wake-up time for duty-cycling with RISS is implemented by the procedure shown in Listing 3.3 .

Listing 3.3: *Estimation of the wake-up time with RISS protocol.*

```
Receive.receive(message)
    sourceAddress = message.getSourceAddress()
    ReceivedMessagesQ2D.enqueue(sourceAddress,message)
    sunTempFreqNumerator = 0
    numberOfMsgProcessed = 0
    //process last N packets from
    //the most recent transmitter
    FOR every message in MessagesQ2D DO
        firstMsgWaitingTxTime = ReceivedMessagesQ2D.
            getMsg(sourceAddress,numberOfMsgProcessed).
                getTxWaitingTime
        secondMsgWaitingTxTime = ReceivedMessagesQ2D.
            getMsg(sourceAddress,numberOfMsgProcessed+1).
                getTxWaitingTime
        IF secondMsgWaitingTxTime − firstMsgWaitingTxTime
```

```
                                        < minWaitTxTime THEN
          minWaitTxTime   =
          secondMsgWaitingTxTime − firstMsgWaitingTxTime
      ENDIF
      firstMsgRxTime = ReceivedMessagesQ2D.
          getMsg(sourceAddress,numberOfMsgProcessed).
              getArrivalTime
      secondMsgRxTime = ReceivedMessagesQ2D.
          getMsg(sourceAddress,numberOfMsgProcessed+1).
              getArrivalTime
      tempFreqNumerator=(secondMsgRxTime−firstMsgRxTime)−
          (secondMsgWaitingTxTime−firstMsgWaitingTxTime)
      sumTempFreqNumerator+=tempFreqNumerator
      numberOfMsgProcessed++
  ENDFOR
  //calculate negihbours clock frequency according to the
      equation 3.13
  avgClockFrequencyNumerator =
      sumTempFreqNumerator/numberOfMsgProcessed
  //calculate future arrival time of the packet from the
  most recent transmitter according to the equation 3.16
  tempFutureArrivalTime = secondMsgRxTime +
          minWaitTxTime + avgClockFrequencyNumerator − G
  SchedulesQ.addFutureArrivalTime(sourceAddress,
                              tempFutureArrivalTime)
  //find next packet arrival time
  scheduleProcessed = 0
  FOR every schedule in SchedulesQ DO
```

```
    IF SchedulesQ.getFutureArrTime(scheduleProcessed) <
                        tempFutureArrivalTime THEN
        tempFutureArrivalTime = SchedulesQ.
                    getFutureArrTime(scheduleProcessed)
ENDFOR
putNodeToSLeep(tempFutureArrivalTime)
```

3.2.5.2 Implementation of RISS for time-stamping of events

As with the implementation of RISS for duty cycling of the nodes, a single application integrates the capabilities of every node used for tests of accuracy of events time-stamping. A diagram showing its main components and their interconnections is presented in Figure 3.7. Some of the modules were used for testing of duty cycling with RISS. Below I only describe those components not presented in the previous section:

- **RISSEventTimeStampingAppC** is the overall application configuration component. It connects the main capabilities of the RISS protocol with other required modules.

- **RISSEventTimeStampingC** is a component that manages the overall application. Depending on the node's ID it decides whether the node is the base station or a subordinate sensor node. It is responsible for managing the RISS protocol, constructing and submitting packets for sending, handling the time-stamping of outgoing packets, and processing received packets. It also handles user button events and sends test results.

- **RISSFreqEstimation** is responsible for calculating the frequency of the neighbour's clock periodically with RISS protocol.

- **AMReceiverAC and AMReceiverBC** are both AMReceiverC generic configurations. This application requires two such components because one is

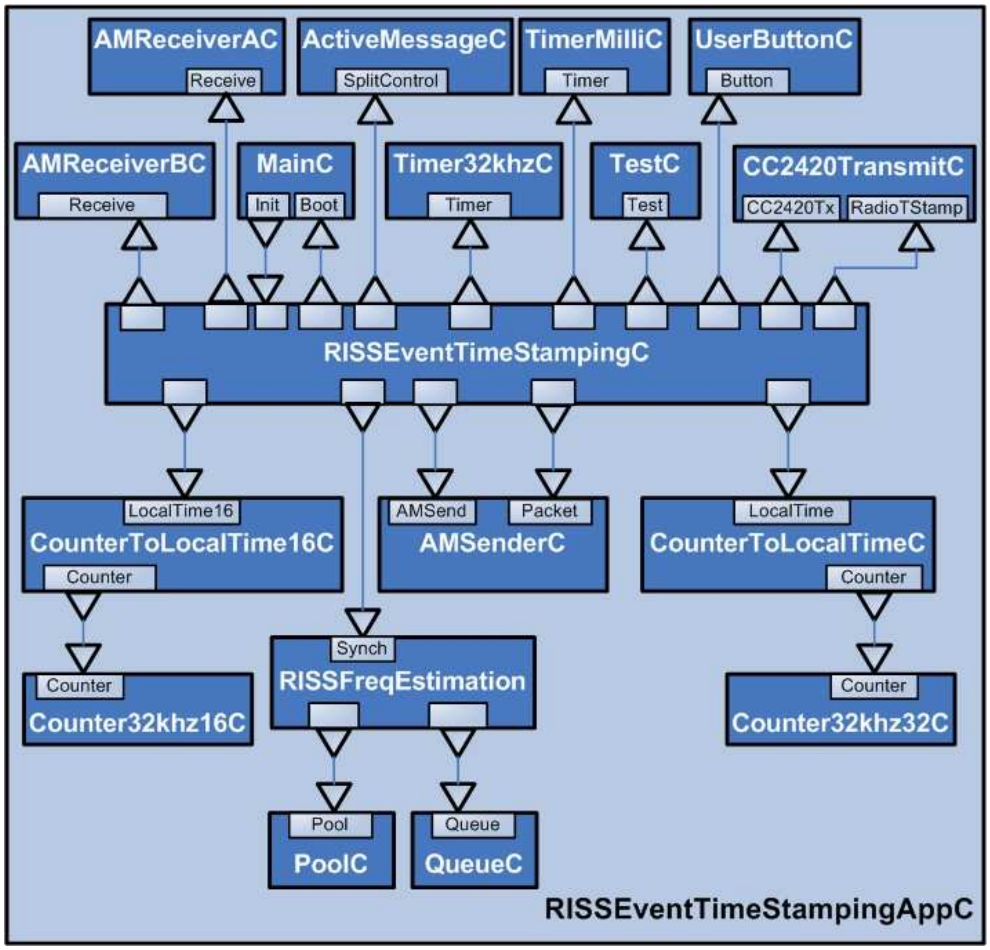


Figure 3.7: *Diagram of the RISS implementation used for event time-stamping - the RISSEvent-TimeStampingAppC configuration. Nodes represent components, and edges represent interface wiring. Triangles are labeled with the corresponding interface name.*

handling AM packets used by the RISS protocol and the other processes packets which simulate an event (see section 3.2.6.4).

- **PoolC** is a general dynamic memory pool component. It provides memory for recording the last Q packets from every neighbour (see section 3.2.4). They need to be stored for estimation of each neighbour's clock frequency.

- **QueueC** is a general FIFO queue component, whose queue has a bounded size. Together with PoolC component it provides a mechanism for storing the last Q packets heard by a node. PoolC provides memory space for an arriving packet whereas QueueC stores a pointer to this memory slot at the end of the queue. Hence the order of arrived packets can be obtained from the QueueC component.

3.2.6 RISS performance

The purpose of my experiments is fourfold:

- to compare the effectiveness of linear regression and fast approximation methods (section 3.2.2.2) to estimate clock frequency

- to optimise the parameters of the RISS protocol

- to verify the robustness of the RISS protocol

- to compare RISS with other synchronisation and duty cycling methods in WSNs

I employed a commonly used sensor network platform, the Tmote Sky sensor node [122], for experiments. Each Tmote Sky node has an 8MHz TI MSP430 microcontroller and a 2.4GHz, 250kbps IEEE 802.15.4 Chipcon wireless transceiver. I built a network composed of four nodes and a single base station. Each node wakes up every 10s, transmits the sensor reading to the base station and goes to sleep. The base station collects 1000 packets from every sensor node. The time of next wake up is estimated with the RISS protocol. The base station also turns off the transceiver when the channel is free. The results of experiments were measured both in hardware and software. At the end of a test I read from software variables the values of measured quantities. I also used a voltage measurement circuit (see appendix A) to estimate the time precision and energy efficiency of the RISS protocol.

3.2.6.1 Comparison of linear regression and fast approximation methods

Initially I compared the computing time of both methods. The results are shown in table 3.2.

	linear regression	fast approximation
time execution	75.6ms	3.6ms
precision	≤ 2 clock ticks	≤ 2 clock ticks

Table 3.2: *Time execution and precision of RISS with linear regression and fast approximation*

Fast approximation method is much faster than linear regression for a similar level of time precision. Processing inefficiency may be a drawback especially if the estimated frequency is needed to duty cycle nodes. A long calculation may significantly reduce the efficiency of that service. Thus I continued the experiments with fast approximation method in order to optimise it.

3.2.6.2 Optimisation of RISS

The performance of the duty cycle service depends on two parameters: Q and G (see equation 3.16) whose optimal values must be determined empirically. In order to do that I built a network composed of four nodes and a single base station. Each node wakes up periodically, transmits the sensor reading to the base station and goes to sleep. The time of the next wake up is estimated with the RISS protocol. The base station also turns off the transceiver when the channel is free. Because the base station estimates the arrival time of the next packet, the overall efficiency of the protocol depends on the performance of the base station.

Optimisation of Q The first series of experiments permits me to estimate the optimal number of past packets (called Q) needed to predict the next packet arrival time (see equations 3.3 and 3.13). There is a trade-off between time execution and precision when choosing Q. As Q increases, the accuracy of estimation increases because a larger number of past packets is considered when estimating next packet's arrival time. However, the execution time of the algorithm (when the node is awake) increases with Q. I want to determine empirically the value of Q that maximises the energy efficiency of the duty cycling protocol. I configured the WSN as follows. Each node sends a sensor sample to the base station every 10s. The base station collects 1000 packets from every sensor node. I measure the average awake time of the base station transceiver as a function of Q. The optimal value of Q is platform dependent and I performed this experiment using TmoteSky nodes [122].

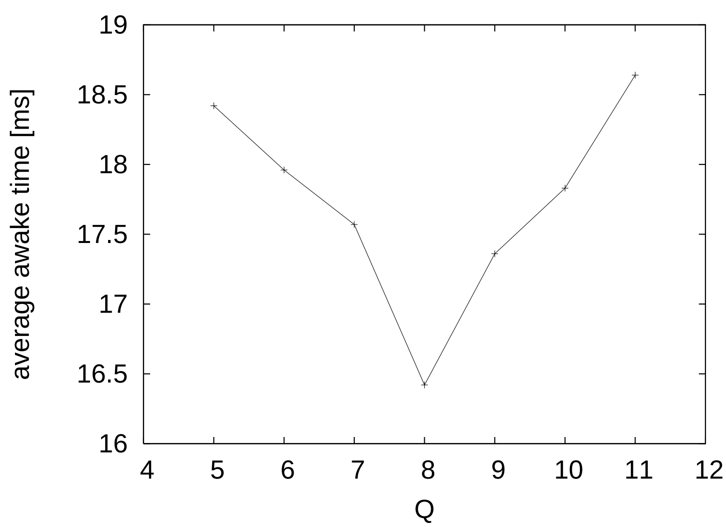


Figure 3.8: *Average transceiver awake time as a function of Q, the number of past packets processed.*

The result of this test is shown in Figure 3.8. The graph has a local minimum when Q equals 8. It means that, for this hardware configuration, the receivers should estimate the arrival of the next packet on the basis of the time-stamps of the 8 previous messages. Then the average time spent listening to the channel, processing the packet and turning off the transceiver is kept low and is approximately 16.5ms.

Optimisation of G Also, I carried out a series of tests to optimise the value of constant G in equation 3.16. It represents the time which I subtract from the estimated time-stamp of the future message and is used in order to compensate for the random component of the arrival time. The energy efficiency drops with the value of G but if I set the constant G too small, a packet loss may occur. This failure of reception disables duty cycling until the arrival of the next packet and is energy inefficient. So, there is a trade off between setting G to a low value which may increase the energy draining because of the missed packets and a larger value of G that extends the awake time for every packet. With the same network of four nodes and a base station I measure the average awake time and number of packets lost as a function of the value of G. The value of Q is set to 8 and every sensor node sends 1000 packets. The results of this experiment are shown in Figure 3.9.

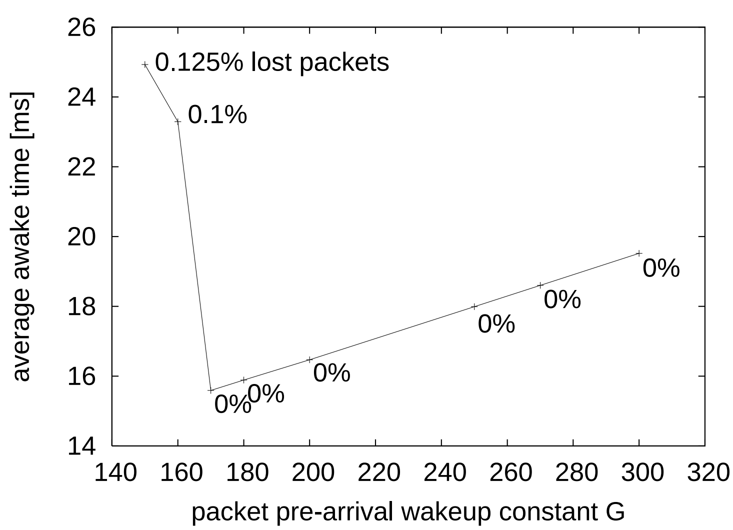


Figure 3.9: *Average transceiver awake time as a function of the pre-awake constant (G in equation 3.16).*

For values of G greater than 170, the average transceiver awake time increases linearly. I do not observe any packet loss in those circumstances. However, if the constant G is below 170, the rate of energy consumption increases. This is due to packets being missed by the receiver. The duty cycling is then disabled until the next message arrival. Four out of 4000 (0.1%) packets missed when G equals 160. When G is 170, the transceiver is powered up $170/32768kHz = 5.19ms$ earlier than predicted in order to compensate for the stochastic component of the packet arrival time.

3.2.6.3 Sensitivity of RISS

In this series of tests I want to verify the influence of application specific parameters on the performance of RISS. I test how the energy efficiency varies with the packet size. I also want to test whether the RISS protocol is sensitive to the duty cycle period.

Influence of packet size on the energy efficiency of RISS With the network of four nodes and a base station, I set up the protocol parameters to the optimal values previously determined. I vary the size of the packet sent by nodes and I record the average awake time of the base station. The processing time of RISS increases linearly with the number of bytes of the message (see Figure 3.10). For the minimum packet size of 4 bytes I measured an awake time of 15.73ms

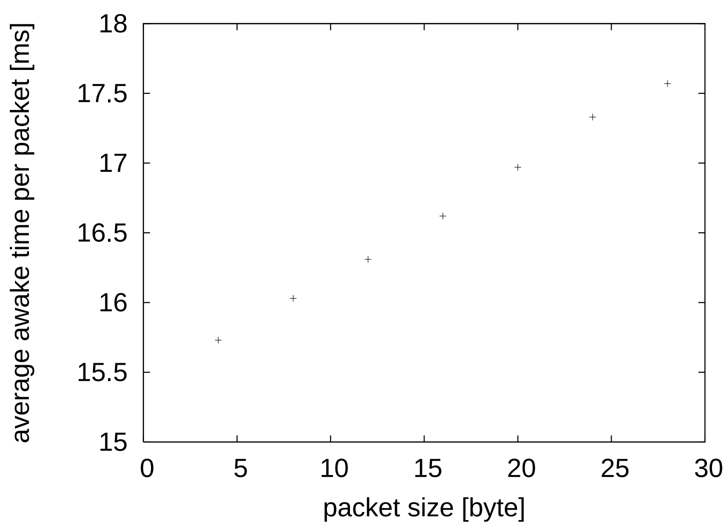


Figure 3.10: *Average transceiver awake time as a function of the packet size.*

and for the Tmote's maximum packet size of 28 bytes I evaluated this value to be

17.57ms. So I deduced that on average the transceiver power up time increases by 0.0784ms for an additional byte in a packet. The CC2420 transceiver has a transmission rate of 250kbps, and so takes 0.032ms to transmit one byte. The difference of 0.0784-0.032=0.0464ms is due to the processing by the duty cycling protocol of an additional byte. Thus in applications where delay is not a major concern, sensor samples should be stored and sent in packets of the maximum size in order to optimise energy efficiency. For example if a node sends a sensor sample of four bytes in an independent packet, the receiver is awakened for 15.73ms to compute the arrival time of the next packet with RISS. However if this sample is inserted in a packet which contains previous sensor samples, it would increase the processing time of the arrival time of the next packet (and thus the awake time) by only 0.18ms.

Influence of the duty cycle period on the energy efficiency of RISS I keep the same network architecture for this test. However, this time I modify the frequency of packets and analyse the variation of the energy efficiency. My observations are plotted in Figure 3.11.

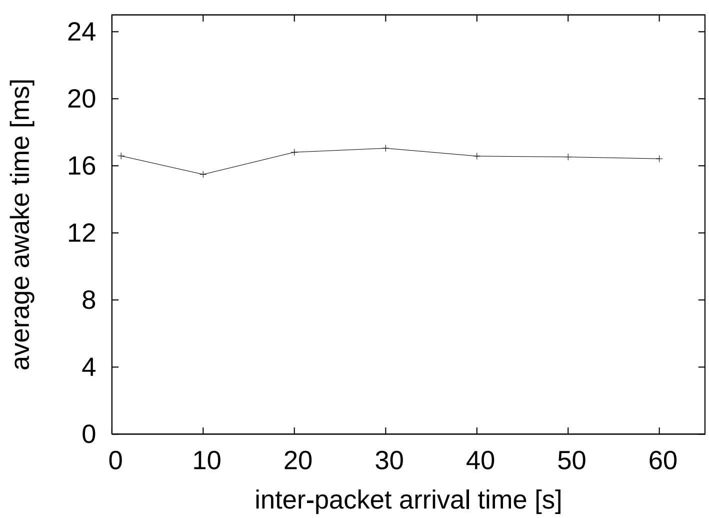


Figure 3.11: *Average transceiver awake time as a function of the inter-packet arrival time*

The energy efficiency is inversely proportional to the average awake time, which from Figure 3.11 can be seen to be relatively insensitive to variations in packet arrival rate. The algorithm performs well in the region where inter-packet arrival time is between one and sixty seconds. Although the performance of RISS may then drop, in real applications we will rarely increase the inter-packet period above one minute because it may lead to a loss of network connectivity, and to a lack of information about the neighbour nodes.

3.2.6.4 Comparison of RISS with other protocols

Comparison of RISS with other duty cycling protocols To compare the duty cycling performance of RISS I chose as a reference the scheduling protocol which is a part of Boomerang, Moteiv's distribution of TinyOS [16]. In the first stage I deployed Boomerang on a network comprising four sensor nodes and a base station. Each node sends a sensor measurement every 10s and the *lowpower* coefficient is set to 1%, the minimum possible value for Boomerang, and thus probably the most efficient. After collecting 500 packets at the base station, I measure the average awake time of every sensor node. In the second phase of the test I installed the RISS protocol on the same network. Similarly, nodes send 500 packets with the inter-packet interval of 10s. The comparative results are plotted in Figure 3.12.

Each node stays asleep for longer when using duty cycling with RISS. The average awake time per node equals 449.6ms with Boomerang and 10.6ms with fast approximation method. Node 0 is the base station. We can observe that with Boomerang it consumes less energy than other nodes. The reverse is the case with RISS protocol (see section 3.2.1). This is due to the different approach to synchronisation. In Boomerang it is the upstream sensor node which adapts to the duty cycling operation of its downstream neighbours. I implemented the opposite solution in RISS (see section 3.2.1). While the difference of average awake time between the base station and the least efficient node is considerable in the case of Boomerang (679.98ms), it is very small (8.06ms) with my protocol. I attribute this to the authors of Boomerang allowing the minimum duty cycle period to be one

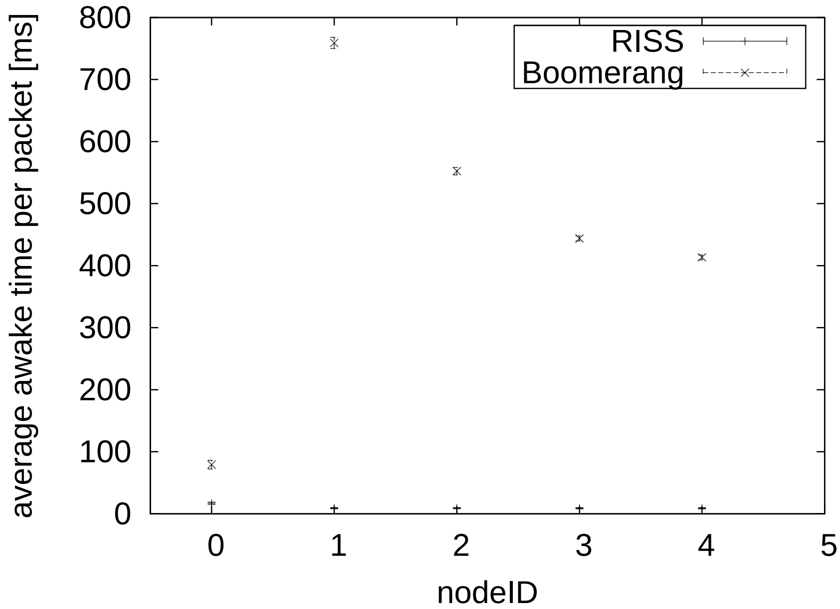


Figure 3.12: *Average transceiver awake time with Boomerang and RISS protocol.*

percent. Also Boomerang requires the exchange of dedicated packets for synchronisation which extends each node's awake time.

Comparison of RISS with other time-stamping protocols The RISS protocol may also be used to determine time-stamp of an event. The performance of the solution can be tested by measuring the difference between the protocol's estimate of an event time and its actual instance. To do this I propose the following experiment scenario. A network is composed of a single node (*node A*) and a base station (*node B*). Node B periodically collects sensor samples from node A. It also estimates node A's clock frequency with the RISS protocol. A node C (the reference broadcaster) broadcasts packets at intervals to nodes B and A. For ease of implementation I assured periodic transmission of these packets. Whenever a packet is heard by the two nodes (A and B), they record the time of occurrence of that packet. When node A sends the sensor reading to node B, it adds to the packet the value of the time which has elapsed (according to A's clock) between the packet broadcast by node C and the SFD of the message being currently transmitted. Then, node B uses that value to calculate the time of reception (according to B's clock) of the packet from node C. It compares the obtained result with the time-stamp it applied to the packet from node C using its local clock upon reception of that packet. The difference between these two values corresponds to an error of the estimation of the arrival time of the packet from the reference broadcaster. For every test I vary the frequency of the sensor packets and I find $\max_i E_i$ where E_i is the error in the i-th iteration of the estimation at the base station of the event time. I compare the precision of RISS with that of FTSP [97] which I tested with the same scenario. I chose FTSP as a reference protocol because it is one of the most prevailing WSN synchronisation protocols and also its implementation on TinyOS is provided by the authors. My observations are shown in Figure 3.13.

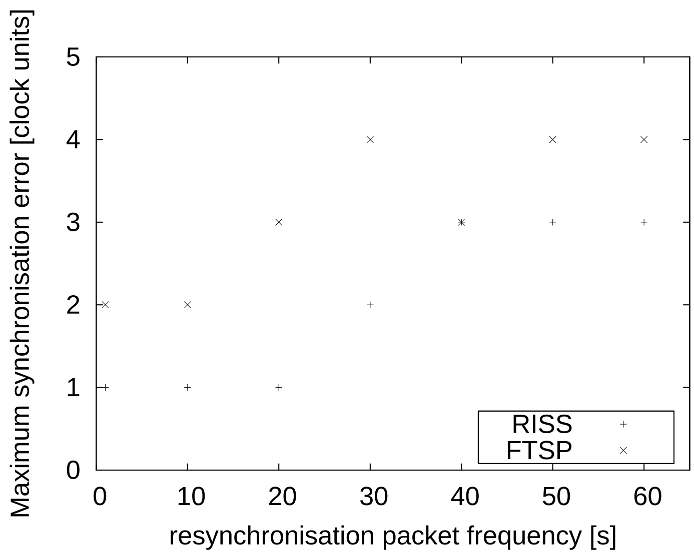


Figure 3.13: *Maximum synchronisation error as a function of the synchronisation beacon frequency for RISS and FTSP protocols.*

On that graph I plotted the observed maximum error in event time estimation for FTSP and RISS. This inaccuracy is expressed as the number of ticks of a 32kHz clock. The exactness of the method depends on the frequency of update messages. This update is done by the exchange of beacon packets in case of FTSP and by the addition of time information to application packets in RISS. The shape of the curve shows that the precision of time synchronisation is a decreasing function of the frequency of the time information exchange between nodes. I observed the smallest maximum synchronisation error of $1/32768 \approx 30.5\mu s$ when the time information was updated every second. Because the maximum synchronisation error can only be expressed as an integer multiple of clock units, FTSP and RISS perform equivalently in terms of synchronisation precision. The difference between the maximum synchronisation error of the two protocols is typically zero or one clock unit. The only exceptions in Figure 3.13 are for the inter-packet periods of 30 and 40s. However, the corresponding maximum synchronisation error of 2 and 3 clock units measured with FTSP protocol was encountered very rarely. Similar outliers would probably be obtained with RISS protocol with a larger set of measurements than the ten used to generate results from Figure 3.13.

3.2.7 RISS: summary

RISS exploits the cyclical operation of the nodes to establish the time reference of every sending neighbour. I showed in previous sections how this information can be used for duty cycling of the nodes and obtaining the time-stamp of sensor samples. The experiments carried in a real environment proved the advantage of RISS over other synchronisation protocols. It performs better than FTSP in terms of synchronisation precision. Also RISS is more energy efficient than Boomerang when used for duty cycling of the nodes. Finally, it is possible to increase significantly the sleeping time of the nodes by sending multiple sensor readings in one packet.

3.3 Summary

The time synchronization problem needs to be addressed in most WSNs. Most WSN applications require time-stamping. Also, the duty cycling of sensor nodes is a common technique to extend the network lifetime. However, the design of the synchronisation service as an independent entity with specific packets, communication scheme and network hierarchy may lead to an inefficient operation of the nodes and to problems such as packet loss. Therefore I have presented two methods for integration of the WSN synchronisation service into the routing layer.

The first method, called Cross Layer Efficient Architecture for Routing (CLEAR), is suitable for networks using a cluster-based architecture. It explores the routing hierarchy to distribute the time information among the nodes and also to reduce the packet loss rate which may be very high in very harsh environments especially due to the limited computational resources of nodes. My experimental results show a significant decrease of the energy dissipation and packet loss when using the CLEAR protocol, compared to existing protocols.

The second method is called Routing Integrated Synchronisation Service (RISS). It is designed to provide time information for nodes in a network with a flat routing architecture. I have demonstrated that RISS is very time and energy efficient and also is characterised by a small overhead. It uses only ten bits per packet to send time information and takes about 16ms to calculate each neighbour's clock frequency using typical sensor nodes. For such a small cost I achieved a higher performance than FTSP. Also the duty cycling service using a fast approximation turned out to be more efficient than the Boomerang implementation of scheduling.

The routing techniques presented in this chapter give an important insight into the cross-layer design approach to routing in WSNs. The experiments show that an integration of synchronisation service into routing layer may lead to efficient scheduling of nodes and accurate time-stamping of events across different network topologies.

CHAPTER 4

Cross-layer routing incorporating location information

Experimental studies have revealed that incorporating information about the location of nodes may increase the energy efficiency of routing protocols in WSNs [128, 152, 162]. If a node only considers those neighbours which are closer to the final destination when forwarding a packet, paths which are not energy efficient can be eliminated. Also WSN sensor nodes communicate over wireless links using a transceiver with regulated output power. Hence, knowledge of node locations enables the choice of a route which minimises the energy dissipation over the path because every node can adapt the transmission power to the minimum required to reach the desired receiver. Finally, the range of a communication link influences the achievable packet reception rate because the signal quality drops with distance from the transmitter. In consequence the transceiver must stay awake for longer (because of the more frequent packet retransmissions) when the receiver is more distant and thus drains more power from the battery. This chapter explores how location information can be used to enhance the energy efficiency of routing protocols.

4.1 SCALE

Frequent wireless transmission of the sensor data can quickly consume a sensor node's energy resources and cause the node to fail. In the case of multi-hop communication, it can lead to the disconnection from the base station of a significant number of sensors for which the failed node was relaying the data. That is why I am not only interested in the minimisation of the overall network energy dissipation but also in the uniform distribution of node lifetimes. This can be achieved by turning off the node's radio modules when not needed. Another possibility is to establish a communication chain depending on the distribution of nodes. Both these operations may be facilitated when the nodes organise themselves into clusters.

In recent years many clustering algorithms have been proposed for WSNs [7, 61, 62, 72, 139] (see section 2.2.4.3). In a clustered architecture sensors send gathered data to the base station through a hierarchy of cluster-heads. These nodes perform such functions as data aggregation, organisation of the transmission schedule or local network management. These operations may drain a lot of energy from a cluster-head. However, the energy expenditure of the overall network can be reduced because they can decrease the number of transmitted packets. In a typical transmission scenario, it has been estimated that 3000 instructions could be executed for the same energy cost as sending a bit 100 meters by radio [123].

In order to explore this property I formulate the transmission path optimisation problem below and present a general optimisation model to determine the intra-cluster topology that maximises the lifetime of the network. Furthermore I propose a variant of the LEACH protocol called Smart Clustering Adapted LEACH (SCALE) which incorporates these results. The improvement proposed can be incorporated into any cluster-based WSN routing protocol. I decided to base my protocol on LEACH because the performance of LEACH compares favourably with other WSNs routing protocols [7, 61, 72]. However my simulations results show that its energy efficiency can be improved by establishing a different intra-cluster topology on the basis of location information concerning each sensor node's neighbours, information which can be learned by cross-layer data sharing.

4.1.1 Problem statement

I consider the problem of cluster formation in order to minimise the overall communication energy consumption in a WSN. The network is composed of randomly distributed nodes. They organise themselves into local clusters, with one node acting as a cluster-head. It collects the data from the nodes in its cluster, aggregates it and transmits to the base station.

The resulting cluster topology can significantly influence the network lifetime. In many proposed routing protocols, nodes join the closest cluster. In consequence when data is sent by a node to the cluster-head it may not necessarily be approaching the base station.

We can prevent this waste of energy by changing the cluster architecture. I analyse the relative position of the base station, the cluster-head and its cluster nodes in order to minimise the communication energy. I want to find the situations when the data traveling through the cluster-head is approaching the base station. I use the path length of direct communication with the base station as a reference and I identify all nodes for which transmission through the cluster-head is more energy efficient than the reference path (Figure 4.1). Then the nodes belonging to that area will form the cluster.

4.1.2 Network and radio models

4.1.2.1 The Network Model

I analyse the relative position and corresponding communication energy between the nodes, the cluster-head and the base station. The nodes are uniformly distributed. I make the conventional assumption that the base station is distant from the network nodes. This assumption is widely made (e.g. [61, 62, 108]). The

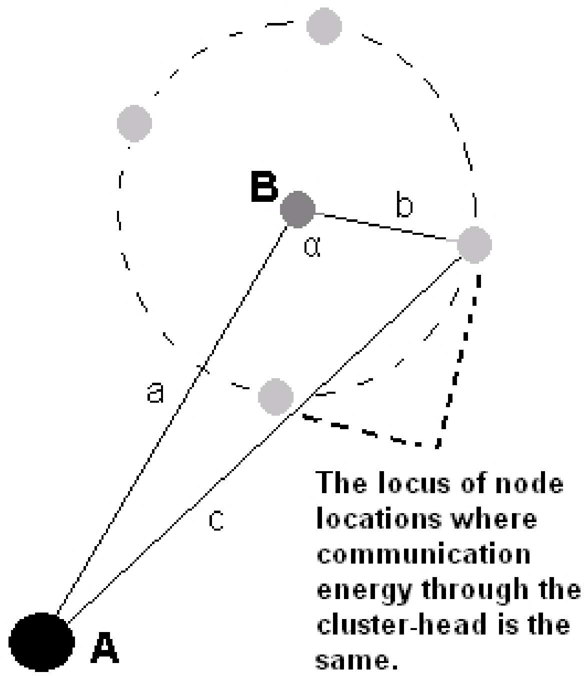


Figure 4.1: *The communication model. From the set of node locations equally distant from the cluster-head I seek those which require less energy for communication through the cluster-head than directly to the base station.*

cluster-head (node B in Figure 4.1) collects the data from the cluster's sensors and transmits it to the base station (node A). The locations of the base station and the cluster-head are fixed. I identify the set of possible node locations for which the energy consumption of the communication through the cluster-head is equal. With my channel model (described later) they can be represented by a circle with the cluster-head at the centre. The cost in energy of communicating with the base station via the cluster-head is unchanged for all locations on the circumference of the circle, but the cost of direct transmission to the base station depends on the angle α (see Figure 4.1). Those angles for which sending data through the cluster-head consumes less energy than direct transmission will now be determined.

4.1.2.2 The Radio Model

Variable	Meaning
r	Distance between transmitter and receiver
$E_T(k,r)$	Transmission energy of k bits over distance r
$E_{T\chi}$	Per-bit energy dissipation during transmission
$E_{amp}(r)$	Per-bit amplification energy
ε_{FS}	Transmit amplifier parameter corresponding to the free-space fading model
ε_{MF}	Transmit amplifier parameter corresponding to the multipath fading model
$E_{R\chi}$	Per-bit energy dissipation of the receiver

Table 4.1: *Variables used in the mathematical formulas.*

To estimate the energy transmission I apply a channel model commonly used in wireless communication [62, 126]. It is a simplified model where the transmitter is characterised by an isotropic radiation and I assume free space propagation (r^2 power loss). However, after a certain threshold distance r_o the multipath model (r^4 power loss) is used to represent the signal propagation. This is because at large distances ($r > r_o$) the multipath fading is a more dominant phenomenon than close to the transmitter. The energy expended when transmitting a single k-bit message over a distance r is expressed in this model by the following equation

$$E_T(k,r) = kE_{T\chi} + kE_{amp}(r) \tag{4.1}$$

The term $E_{T\chi}$ denotes the per-bit energy dissipation during transmission. $E_{amp}(r)$, the per-bit amplification energy, is proportional to r^4 (the multipath fading) when the transmission distance exceeds the threshold r_o and otherwise is proportional to r^2 (the free space model). $E_{amp}(r)$ is thus given by

$$E_{amp}(r) = \begin{cases} \varepsilon_{FS} r^2, & r \le r_o \\ \varepsilon_{MF} r^4, & r > r_o \end{cases} \tag{4.2}$$

The parameters ε_{FS} and ε_{MF} denote transmit amplifier parameters corresponding to the free-space and the multipath fading models respectively. They depend on the required sensitivity and the receiver noise figure, as the transmit power needs to be adjusted so that the power at the receiver is above certain minimum threshold. The value of r_o is given by $r_o = \sqrt{\varepsilon_{FS}/\varepsilon_{MF}}$. The reception energy of the k-bit data message can be expressed by the equation:

$$E_R(k) = kE_{R\chi} \tag{4.3}$$

where $E_{R\chi}$ is the per-bit energy dissipation of the receiver. In the theoretical analysis I neglect this term and $E_{T\chi}$. This assumption is made in many publications [45, 102, 134, 137]. The effect on the results obtained is negligible which I verified with simulations for which I incorporate the reception energy and the per-bit energy dissipation during transmission ($E_{T\chi}$). For the simulations I apply the numerical values of the communication parameters used for the simulations of LEACH by Heinzelman *et al.* [61]: $E_{T\chi} = E_{R\chi} = 50$ nJ/bit, $\varepsilon_{MF} = 0.0013$ pJ/bit/m^4, $\varepsilon_{FS} = 10$ pJ/bit/m^2.

4.1.3 Theoretical analysis

From the set of candidate nodes equidistant from the cluster-head, we select those which offer better energy efficiency when communicating through the cluster-head in comparison to direct transmission to the base station. This (from Figure. 4.1) corresponds to the constraints:

$$E_T(k,b) + E_T(k,a) < E_T(k,c)$$

or equivalently

$$E_{amp}(b) + E_{amp}(a) < E_{amp}(c) \tag{4.4}$$

The expression $E_{amp}(r)$ depends on the value of distance r and thus the inequality 4.4 can result in eight cases to analyse.

$$\varepsilon_{_{FS}} b^2 + \varepsilon_{_{FS}} a^2 < \varepsilon_{_{FS}} c^2 \text{ when } a, b, c \leq r_o \tag{4.5}$$

$$\varepsilon_{_{MF}} b^4 + \varepsilon_{_{FS}} a^2 < \varepsilon_{_{FS}} c^2 \text{ when } a, c \leq r_o \text{ and } b > r_o \tag{4.6}$$

$$\varepsilon_{_{FS}} b^2 + \varepsilon_{_{MF}} a^4 < \varepsilon_{_{FS}} c^2 \text{ when } b, c \leq r_o \text{ and } a > r_o \tag{4.7}$$

$$\varepsilon_{_{MF}} b^4 + \varepsilon_{_{MF}} a^4 < \varepsilon_{_{FS}} c^2 \text{ when } c \leq r_o \text{ and } a, b > r_o \tag{4.8}$$

$$\varepsilon_{_{MF}} b^4 + \varepsilon_{_{MF}} a^4 < \varepsilon_{_{MF}} c^4 \text{ when } a, b, c > r_o \tag{4.9}$$

$$\varepsilon_{_{FS}} b^2 + \varepsilon_{_{FS}} a^2 < \varepsilon_{_{MF}} c^4 \text{ when } a, b \leq r_o \text{ and } c > r_o \tag{4.10}$$

$$\varepsilon_{_{MF}} b^4 + \varepsilon_{_{FS}} a^2 < \varepsilon_{_{MF}} c^4 \text{ when } a \leq r_o \text{ and } b, c > r_o \tag{4.11}$$

$$\varepsilon_{_{FS}} b^2 + \varepsilon_{_{MF}} a^4 < \varepsilon_{_{MF}} c^4 \text{ when } b \leq r_o \text{ and } a, c > r_o \tag{4.12}$$

Inequality 4.5 describes the situation when all transmissions are represented by the free-space model. We can find the angles α which realise this inequality by a geometrical analysis. For any triangle we can write:

$$c^2 = b^2 + a^2 - 2ab\cos\alpha \tag{4.13}$$

Substituting for c^2 in inequality 4.5 and eliminating $\varepsilon_{_{FS}}$, reduces it to

$$2ab\cos\alpha < 0 \tag{4.14}$$

Thus inequality 4.5 applies for $\alpha \in (\pi/2; 3\pi/2)$.

Inequalities 4.6, 4.7 and 4.8 will not be realised in practice. Inequality 4.6 corresponds to a situation when $c < b$ (because it applies when $c \leq r_o$ and $b > r_o$). It means that a node transmits to a cluster-head which is further away than the base station. This is obviously less energy efficient than direct communication with the base station. Similar reasoning applies for inequality 4.7, although in this case the cluster head is more distant from the base station than the node. Sending packets through this cluster-head is evidently more energy demanding than direct communication with the base station. Inequality 4.8 corresponds to the situation when both the distances between the node and the cluster-head, and that between the cluster-head and the base station are larger than the distance between the node and the base station. This situation is again not relevant in practice.

To solve **inequality 4.9** I square equation 4.13 and substitute for c^4 in the inequality 4.9. This results in an inequality of the second degree (inequality 4.15) with an unknown $\cos\alpha$.

$$4a^2b^2\cos^2\alpha - 4ab(a^2+b^2)\cos\alpha + 2a^2b^2 > 0 \tag{4.15}$$

This may be solved for $\cos\alpha$ to obtain

$$\cos\alpha < \frac{a^2+b^2-\sqrt{a^4+b^4}}{2ab} \qquad \text{or} \qquad \cos\alpha > \frac{a^2+b^2+\sqrt{a^4+b^4}}{2ab} \tag{4.16}$$

The second condition is never realised because $\frac{a^2+b^2+\sqrt{a^4+b^4}}{2ab}$ is greater than one for positive a and b. In the first condition, the term $\frac{a^2+b^2-\sqrt{a^4+b^4}}{2ab}$ is always positive. So the angles that realise that condition are from the interval $(\beta; 2\pi-\beta)$ and $\beta \in (0; \pi/2)$.

Similarly, to solve **inequality 4.10** I square equation 4.13 and substitute for c^4 in inequality 4.10. This results in an inequality of the second degree (inequality 4.17) with an unknown $\cos\alpha$.

$$\varepsilon_{MF}4a^2b^2\cos^2\alpha - \varepsilon_{MF}4ab(a^2+b^2)\cos\alpha + \varepsilon_{MF}(a^2+b^2)^2 - \varepsilon_{FS}(a^2+b^2) > 0 \tag{4.17}$$

This requires

$$\cos\alpha < \frac{\varepsilon_{MF}(a^2+b^2) - \sqrt{\varepsilon_{MF}\varepsilon_{FS}(a^2+b^2)}}{\varepsilon_{MF}2ab} \quad \text{or} \quad \cos\alpha > \frac{\varepsilon_{MF}(a^2+b^2) + \sqrt{\varepsilon_{MF}\varepsilon_{FS}(a^2+b^2)}}{\varepsilon_{MF}2ab}$$

The second condition is never realised because $\frac{\varepsilon_{MF}(a^2+b^2)+\sqrt{\varepsilon_{MF}\varepsilon_{FS}(a^2+b^2)}}{\varepsilon_{MF}2ab}$ is greater than one for positive a and b. In the first condition, the term $\frac{\varepsilon_{MF}(a^2+b^2)-\sqrt{\varepsilon_{MF}\varepsilon_{FS}(a^2+b^2)}}{\varepsilon_{MF}2ab}$ is always positive. Furthermore, for relevant values of ε_{FS} and ε_{MF} (see section 4.1.2.2) it is greater than $\frac{a^2+b^2-\sqrt{a^4+b^4}}{2ab}$. So the angles that realise that condition are from the interval $(\gamma; 2\pi-\gamma)$ and $\gamma \in (0;\beta)$.

Inequality 4.11 cannot be realised in a real WSN because in this case $a < b$ (since $a \le r_o$ and $b > r_o$). It means that the node is further from the cluster-head than the base station. Such a situation is impossible in the defined network configuration where the base station is distant from the nodes in comparison to the inter-node distances.

To solve **inequality 4.12** I square equation 4.13 and substitute for c^4 in inequality 4.12. This results in an inequality of the second degree (inequality 4.18) with an unknown $\cos\alpha$.

$$\varepsilon_{MF}4a^2b^2\cos^2\alpha - \varepsilon_{MF}4ab(a^2+b^2)\cos\alpha + \varepsilon_{MF}(b^4+2a^2b^2) - \varepsilon_{FS}b^2 > 0 \tag{4.18}$$

This inequality is satisfied when

$$\cos\alpha < \frac{\varepsilon_{MF}(a^2+b^2) - \sqrt{\varepsilon_{MF}(\varepsilon_{MF}a^4+\varepsilon_{FS}b^2)}}{\varepsilon_{MF}2ab}$$
or
$$\cos\alpha > \frac{\varepsilon_{MF}(a^2+b^2) + \sqrt{\varepsilon_{MF}(\varepsilon_{MF}a^4+\varepsilon_{FS}b^2)}}{\varepsilon_{MF}2ab} \tag{4.19}$$

The second condition is never realised because $\frac{\varepsilon_{MF}(a^2+b^2)+\sqrt{\varepsilon_{MF}(\varepsilon_{MF}a^4+\varepsilon_{FS}b^2)}}{\varepsilon_{MF}2ab}$ is greater than one for positive a and b. In the first condition, the value of the term $\cos\alpha <$

$\frac{\varepsilon_{MF}(a^2+b^2)-\sqrt{\varepsilon_{MF}(\varepsilon_{MF}a^4+\varepsilon_{FS}b^2)}}{\varepsilon_{MF}2ab}$ depends on distances a and b. The angles that realise that condition are from the interval $(\delta; 2\pi - \delta)$ where δ belongs to $(0; 2\pi)$.

To summarise, inequality 4.4 is satisfied when

a) $\alpha \in (\pi/2; 3\pi/2)$ for $a, b, c \leq r_o$

b) $\alpha \in (\beta; 2\pi - \beta)$ and $\beta \in (0; \pi/2)$ for $a, b, c > r_o$

c) $\alpha \in (\gamma; 2\pi - \gamma)$ and $\gamma \in (0; \beta)$ for $a, b \leq r_o$ and $c > r_o$

d) $\alpha \in (\delta; 2\pi - \delta)$ and δ belongs to $(0; 2\pi)$ and $\delta = f(a, b)$ for $b \leq r_o$ and $a, c > r_o$

These conditions are illustrated in Figure 4.2.

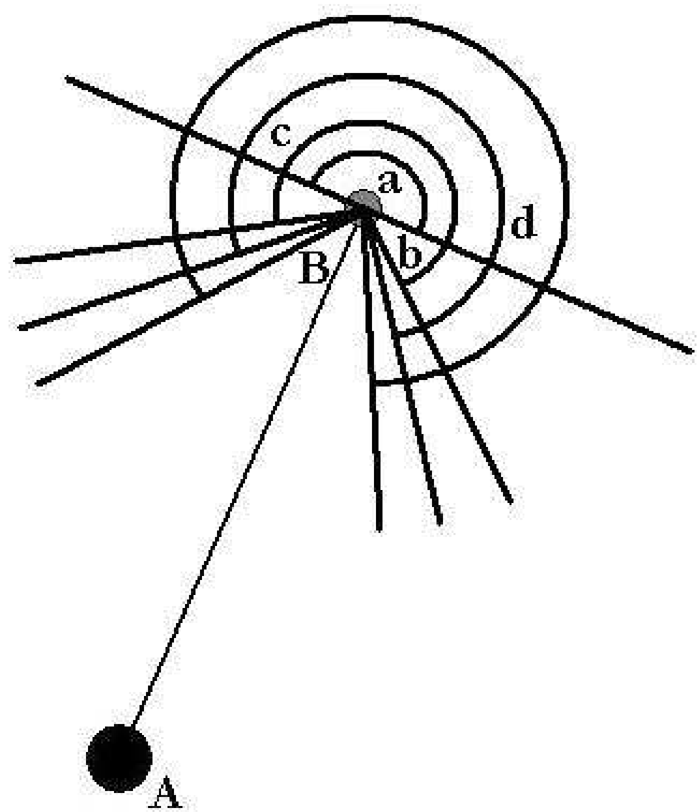


Figure 4.2: *Possible solutions of the inequality 4.4. a)* $\alpha \in (\pi/2; 3\pi/2)$ *for* $a, b, c \leq r_o$*, b)* $\alpha \in (\beta; 2\pi - \beta)$ *and* $\beta \in (0; \pi/2)$ *for* $a, b, c > r_o$*, c)* $\alpha \in (\gamma; 2\pi - \gamma)$ *and* $\gamma \in (0; \beta)$ *for* $a, b \leq r_o$ *and* $c > r_o$*, d)* $\alpha \in (\delta; 2\pi - \delta)$ *and* δ *belongs to* $(0; 2\pi)$ *and* $\delta = f(a, b)$ *for* $b \leq r_o$ *and* $a, c > r_o$

A subordinate node should select the best available cluster-head using the above criteria. In practice, verification of these conditions would require a substantial knowledge about network topology. It would cost a considerable amount in time and energy to acquire that information. Thus I propose to reduce the number of criteria to a single condition. I select a condition which is true in all four cases and figure 4.2 shows that when criterion a) is satisfied, conditions b) and c) are as well. Thus we can ignore the latter two criteria. Hence I use criterion a) but determine the values of a and b for which condition d) reduces to condition a).

When $\delta < \pi/2$, $\cos\alpha < x$ and $x > 0$. Thus from equation 4.19 we can deduce that

$$\frac{\varepsilon_{MF}(a^2+b^2)-\sqrt{\varepsilon_{MF}(\varepsilon_{MF}a^4+\varepsilon_{FS}b^2)}}{\varepsilon_{MF}2ab} > 0 \tag{4.20}$$

$$a^2 > \frac{\varepsilon_{FS}}{2\varepsilon_{MF}} - \frac{b^2}{2} \tag{4.21}$$

We can assume that $b \ll a$. Substituting the numerical values of ε_{MF} and ε_{FS} used by Heinzelman *et al.* [61] we obtain:

$$a > 62m \tag{4.22}$$

In recapitulation, when the distance between the source and destination node is over 62 meters then the communication through an intermediate node is energy saving provided the angle at the intermediate node (angle α in the Figure 4.1) is between 90^o and 270^o. This situation is represented in Figure 4.3.

4.1.4 The SCALE protocol - an enhancement of LEACH

Based on the result of the previous section I propose a modification of the cluster formation phase in the LEACH protocol. As stated in section 4.1.1, in many routing protocols (for example in LEACH) a node joins the closest cluster-head. I am looking at the overall chain of communication, and so the direction as well

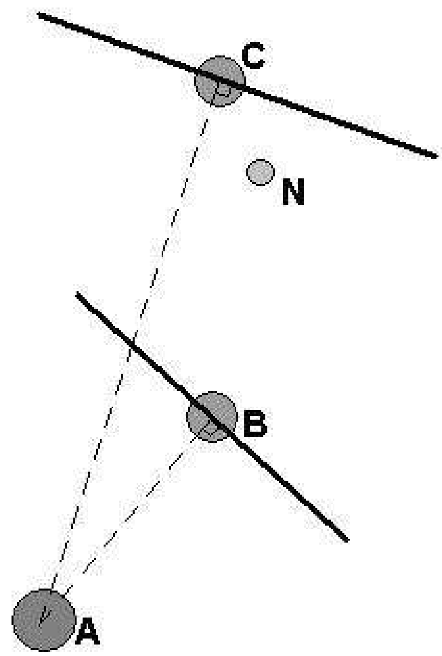


Figure 4.3: *Node N will join cluster-head B because the angle formed by N, B and A is between* 90^o *and* 270^o.

as the distance of data transmission is important. From a study of inequality 4.4 I conclude that the network lifetime can be prolonged if the angle between the base station, cluster-head and a node is in the range $(\pi/2;3\pi/2)$. Directly estimating the angle of arrival for such a small device as the sensor node is problematic. I propose to estimate the angle using distance data.

In my protocol, called the Smart Clustering Adapted LEACH (SCALE) I assume that each node knows the distance separating it from the base station. It can be determined using location information obtained with any WSN localisation protocol. Now the sensor has all the information necessary to calculate the angle α. It searches the cluster-heads for which $a^2 + b^2 < c^2$ (in Figure 4.1) because in this case the angle α is between π and $3\pi/2$ radians. It selects the closest of these nodes as its cluster-head in the following round. If there is no such cluster-head, the node joins the closest cluster, as in the basic LEACH protocol.

4.1.5 Experimental results

To validate my cluster formation scheme, I simulated the LEACH and SCALE protocols using Java. The random 100-node network used for the simulation is shown in Figure 4.4. The base station is located at the co-ordinate (200, 200) off the figure. Every node is initially given 2J of energy. The transmission energy was calculated using equation 4.1. Each node transmits one packet every minute with a length of 20 bits to the cluster-head. Before retransmitting the data to the base station, the cluster-head aggregates it. This aggregation can be simulated in different ways. One approach is to multiply the number of bits received from the nodes in a cluster by a coefficient c smaller than 1. I compared efficiency of SCALE for various values of c. We can also aggregate the received packets into one. This scenario was also represented in my tests.

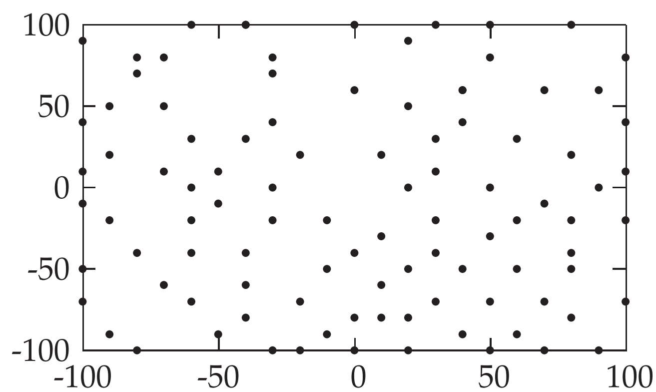


Figure 4.4: *100-node random network*

SCALE performance compared to LEACH I am comparing the mean energy being drained from the nodes for both protocols. I examine also the network lifetime which I define to be the time of the first failure of any node caused by lack of energy. This approach has been followed by many authors [61, 62, 141].

In the first simulation I do not implement any data aggregation at the cluster-head. The probability of becoming a cluster-head is set to 10%. I run the simulation until the first node dies which happens for LEACH at about round 330.

As can be seen in Figure 4.5, the difference in average energy consumption of the two protocols increases with time, reaching 20% when the first node fails using LEACH. Figure. 4.6 shows the network lifetime for both the LEACH and

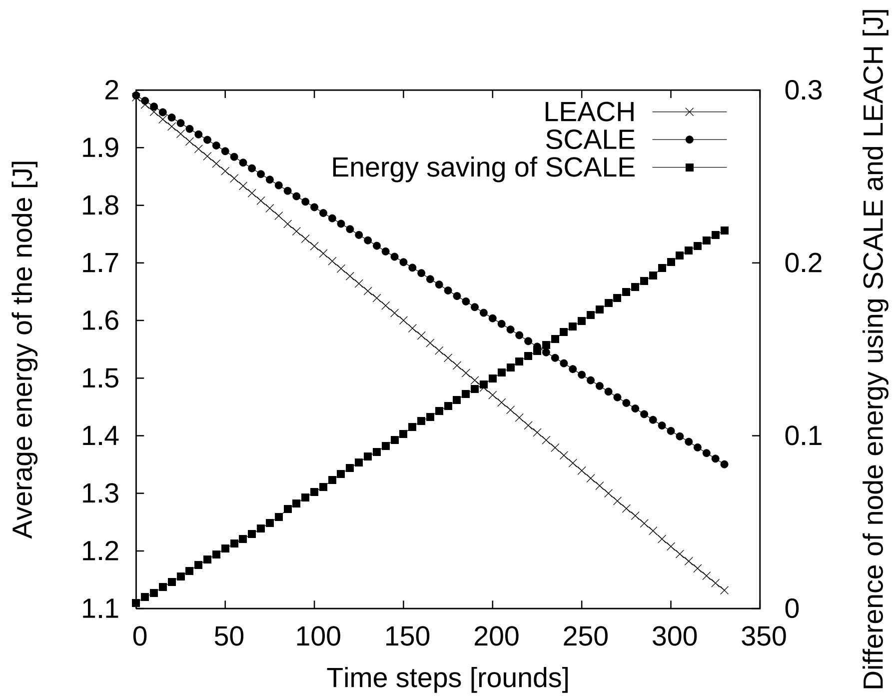


Figure 4.5: *Average node energy with LEACH and SCALE protocols.*

SCALE protocols. In the former case the first node dies after 330 rounds and in the latter after 546 rounds which represents an improvement of 65%. However, by about round 1300 more nodes die with the SCALE protocol. This happens because compared to LEACH, SCALE drains energy faster from the nodes closer to the base station as they have to retransmit packets from more nodes when they become cluster-head. However for both protocols the nodes distant from the base station die first. Therefore after the majority of remote nodes have failed the behaviour of both protocols converges. This is because for remote nodes it is more probable with SCALE to find a cluster-head in the direction of the base station (*i.e.* with the angle α at the cluster-head between $\pi/2$ and $3\pi/2$ radians). When a node cannot find such a cluster-head, it chooses the closest cluster-head (similarly to the LEACH protocol). Since this case applies more often to nodes close to the base station than to those far away and since the latter nodes will fail sooner

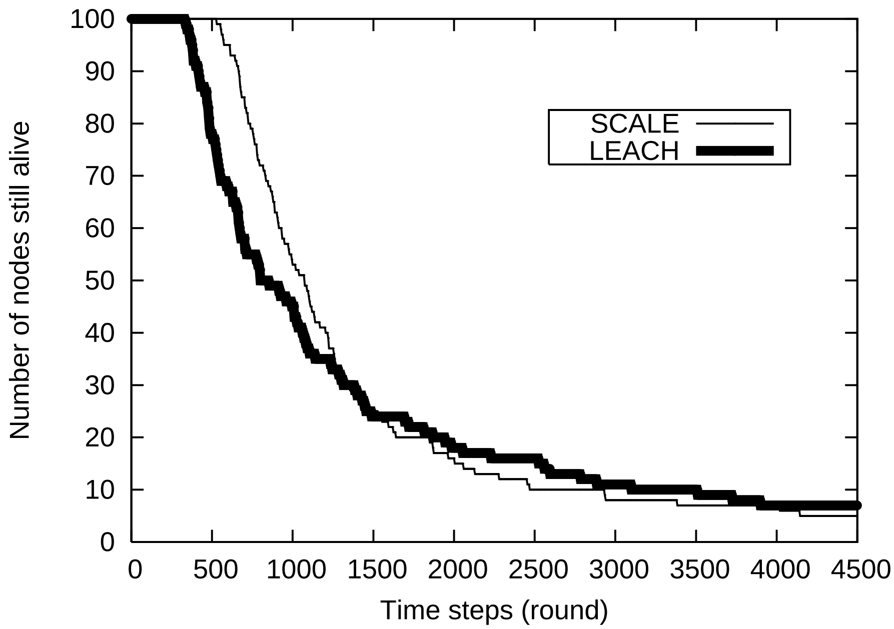


Figure 4.6: *System lifetime using LEACH and SCALE protocol.*

due to their higher power consumption, the discrepancies in the behaviour of the two protocols will diminish over time. The remaining nodes have less energy resources when they operate with SCALE because as mentioned above this protocol drains more energy from the nodes closer to the base station than LEACH. In consequence these nodes will be dying at a faster rate than if they operated with the original LEACH protocol from the beginning.

SCALE performance as a function of the distance between the network and base station The performance improvement of SCALE over LEACH is particularly pronounced when the base station is far away from the majority of cluster-heads (see Figure 4.7).

This is because when the base station is moved farther away this influences the communication cost between the cluster-heads and base station rather than between the subordinate nodes and cluster-heads, this path loss being a power function of the distance between source (remote cluster-head) and destination (base station). In the LEACH case these remote cluster-heads tend to communicate with more sensors than in the SCALE protocol. Hence base station remoteness will

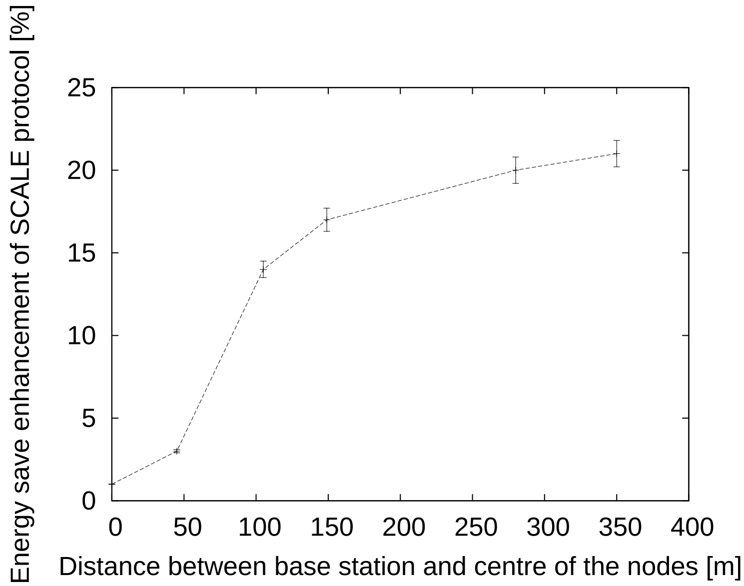


Figure 4.7: *SCALE energy saving compared to the LEACH.*

have a more adverse influence on the energy consumption in the LEACH protocol. Besides, the situation when the base station is more remote favours the communication of subordinate nodes with larger number of edge cluster-heads. Since more cluster-heads are available for communication with a remote base station in SCALE, energy is more uniformly drained from them and thus the overall network lifetime is increased. This is shown in Figure 4.8.

SCALE performance as a function of the compression ratio The main benefit from a clustered protocol comes from data aggregation at cluster-heads – the cluster-head gathers information from its subordinate nodes and aggregates it into one packet. The data reduction achieved depends on the correlation of the sensor measurements. I model this using a compression level parameter C. The total data transmitted following aggregation is assumed to be $D_A = (1 - C)D_R$ where D_R is the amount of raw data (in bits). Figure 4.9 shows the lifetime difference between the LEACH and SCALE protocols. The probability of becoming a cluster-head was set to 10% for the simulations, so a compression level of 0.9 corresponds to the situation when the data generated by the cluster during one "cycle of operation" is aggregated into a single packet at the cluster-head.

The performance advantage of SCALE compared to LEACH diminishes with

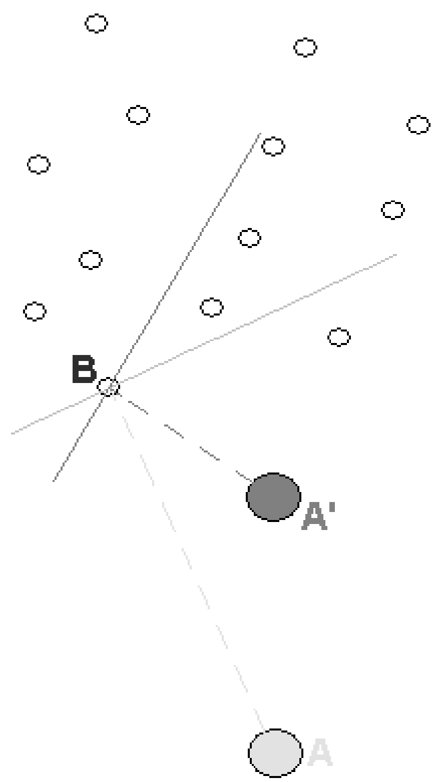


Figure 4.8: *When the base station is in the location A, a node B is eligible to become a cluster-head for more nodes than when the base station is located in A′.*

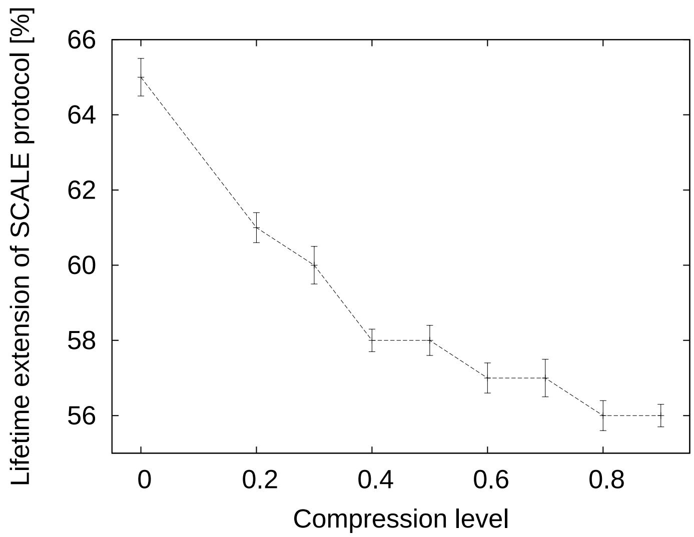


Figure 4.9: *SCALE lifetime compared to the LEACH.*

increasing data compression level. In the LEACH protocol data is transmitted in two phases: from the source to the cluster-head and from the cluster-head to the base station. The SCALE protocol reduces the communication distance of the latter phase at the price of increased space between the sensor and its cluster-head. The data compression decreases the communication energy of the second phase. Because this phase is more dominant in the original LEACH protocol, changing the compression ratio will have more influence on the overall communication cost for this protocol.

4.1.6 SCALE: summary

In the previous sections I have studied how cluster formation in WSNs affects overall network communication energy. I have proposed the angle between the node, cluster-head and the base station as a criterion for choosing cluster-heads. This method requires a cross-layer design approach to share information about node locations. I employ this approach in a new protocol called SCALE. Simulations show that SCALE performs better in terms of energy consumption and system lifetime than the LEACH protocol.

4.2 Hop-distance influence on energy efficiency of a route

A fundamental design choice in WSN routing concerns whether packets should be sent over many short hops or over a smaller number of longer hops. Recently this debate has drawn significant research attention [39, 52–54, 116, 156].

The debate over the number of required hops comes from the fact that each strategy (long-hop and short-hop routing) has its own advantages. Routing over many short hops minimises the transmission energy which increases with the communication distance. However, sending packets over long distance relays reduces the reception cost (as the number of nodes involved in data routing decreases). This section analyses the impact of the hop distance on network lifetime with the following assumptions:

- I consider WSNs of large density with many nodes located between source and the base station. Such a topology can be encountered with WSNs used for monitoring the environment, offices, systems or industrial sites.

- To compare the energy efficiency of different schemes, I use the Minimum Energy Routing (MER) metric which incorporates the total energy consumed on a path [105].

I propose a method of determining the optimal location of the relay node (the node through which data is routed in a two-hop network). This scheme can be used in a routing protocol by applying a cross-layer design approach where location information of the nodes is shared among communication stack layers.

4.2.1 Problem statement

4.2.1.1 Applications considered

Numerous possible applications have been proposed for WSNs. They can be used for monitoring the environment, habitat, offices, systems, buildings, and industrial sites. They can help rescue teams gather information about dangerous situations or people in need of assistance. They can be included as part of a patient's monitoring system in hospitals. There are many other situations where WSNs could be employed. However in this section I wish to focus on the first mentioned group of applications. Buildings, sites, structural, habitat and environmental monitoring represent a very broad class of sensor network usage with enormous potential benefits for scientific communities and industry. These applications will dictate the type of network topology I have to examine.

4.2.1.2 Network topology

In systems monitoring the environment or offices, WSNs are generally deployed over large areas with hundreds or thousands of nodes. Sensors collect the information periodically and transmit it to the one location called the base station. This communication scheme limits the number of established links, because for every node the final destination is always the base station. As the network is very dense, we can assume that there is a large number of relays aligned along the path joining the source to the destination. In the previously mentioned applications, nodes are immobilised and they send a report about monitored parameters periodically to the base station. The network protocol overhead is very small because once the communication architecture is established, it can last for a long time (depending on the routing strategy undertaken). Its energy efficiency may be evaluated using many metrics.

4.2.1.3 Energy Efficiency

Some WSN routing protocols try to send packets over the longest possible hops to reduce the overall "linkload" between nodes. Others may implement the short-hop strategy because of the significant signal attenuation. Either strategy can be advantageous and hence we need a metric to compare the energy efficiencies of the long-hop and short-hop scheme. We can use one of three criteria of evaluation. The first is called Minimum Transmission Energy (MTE), and involves comparing only the energy used for the transmission by every node. Another is the Minimum Energy Routing (MER) which incorporates the total energy consumed on a path. The final metric, called Maximum Network Lifetime (MNL) considers the lifetime of the network which represents the time until the first depletion of a node's battery. I choose the MER metric for the following reasons.

- The specifications of some WSNs transceivers show that reception cost can be even higher than transmission cost (Figure 4.10).
- MNL seeks to maximise the lifetime of the first node to fail. However, in the applications considered here, the system may survive the failure of a proportion of the nodes.
- The strategy of seeking to maximise network lifetime may be inappropriate in WSNs because it requires a significant protocol overhead and in case of nonuniform distribution of energy resources of the nodes, the cost of prolonging the network lifetime can be unreasonable.

4.2.1.4 Questions to address

I consider very dense WSNs with many potential relays to the base station. In such cases, a short-hop routing scheme is never the most advantageous when using the MER metric. This is mainly caused by the significant energy consumption during reception in some existing WSNs (see Figure 4.10) which may be even

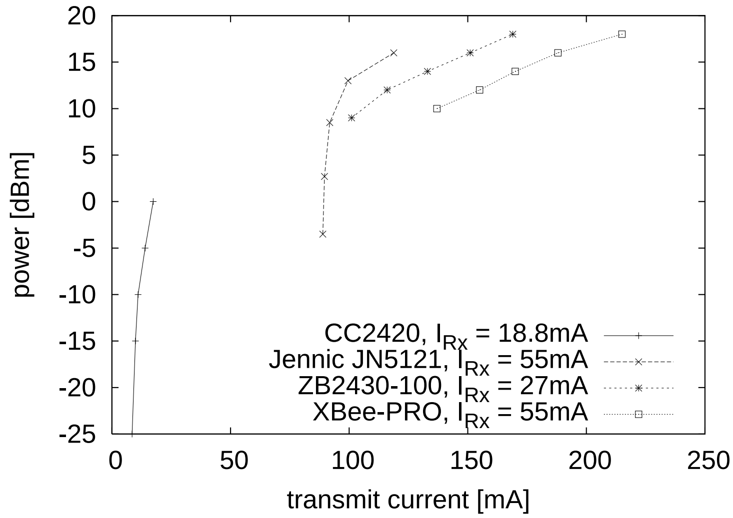


Figure 4.10: *Current consumption in transmission (for different RF power levels) and reception (I_{Rx}) of existing WSN modules: CC2420 (voltage supply 3V), XBee-PRO (3.3V), Jennic JN5121-000-M02 (3V), ZB2430-100 (3.3V).*

higher than when the node is transmitting. As a consequence I would like to find out whether and when the long-hop strategy is optimal using the MER metric. With this scheme a large portion of nodes can communicate directly to the base station since the outdoor transmission range can attain 1200m [26]. Hence I consider when the nodes should relay packets through other sensors or send them to the final destination. I want also to know which two-hop topology is optimal. I examine those questions first with an analytical study.

4.2.2 Analytical study

I approach the multi-hop energy efficiency problem in the following way. First, I describe the wave propagation model used in my study. Next, I explain why we should compare the two-hop transmission with direct communication in order to deduce when multi-hop routing is advantageous using the MER metric. Then, I study the total power radiated using both schemes: single and multi-hop sending. In the next step, I determine the relationship between the power radiated by the node and the energy drained from the battery. Finally, I determine when each of the strategies considered should be applied.

4.2.2.1 Propagation model

In my study I use the log-distance path loss model described in section 2.4.1. The power received by a node a distance of d meters from the sender can be expressed as follows:

$$P(d) = P_0 \times \left(\frac{d_0}{d}\right)^{\alpha} \tag{4.23}$$

where P_0 represents the power of the signal received at distance d_0 from the source and α is the path loss exponent. The value of α depends on the specific propagation environment and experiments have shown that it usually takes a value between 2 and 5 [24, 126]. I use equation 4.23 to express the minimum power required to communicate over a given distance and I compare the two routing strategies.

4.2.2.2 Multi-hop vs Single-hop problem reformulation

When the nodes are communicating with the minimum power necessary to reach the destination, if we consider only the total power transmitted over the path, then the short-hop strategy would be the most energy efficient. This is caused by the signal attenuation which is proportional to the power function (equation 4.23). However, since the reception cost should not be neglected, there is a minimum range of source-destination distance for which direct communication is an optimal alternative. This is because the savings in transmission power by the multi-hop scheme does not compensate for the resulting additional reception energy cost. If we enlarge the distance between source and base station, there is a bound for which the two-hop routing becomes more advantageous. If the transmission distance increases, the 3-hop communication will become optimal and so forth. Therefore if we wish to know which routing strategy is more energy efficient for a given topology then we need to compare the energy consumption of single-hop and two-hop transmission. Figure 4.11 represents the two-hop and di-

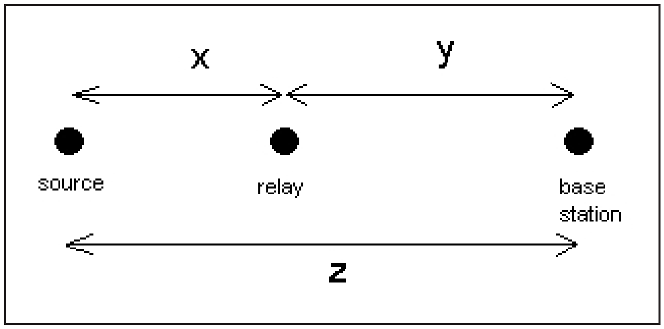


Figure 4.11: *One-hop and direct transmission.*

rect transmissions. Every node is transmitting with the minimum power required to guarantee that the signal at the receiver is above the sensitivity level P_m. For each link we can therefore write:

$$P_m = P_x \times \left(\frac{d_0}{x}\right)^{\alpha} = P_y \times \left(\frac{d_0}{y}\right)^{\alpha} = P_z \times \left(\frac{d_0}{z}\right)^{\alpha} \tag{4.24}$$

We can now compare when two-hop routing is more energy efficient:

$$P_x + P_y + P_r < P_z \tag{4.25}$$

where P_r represents the power drained at the reception from the relay. If we substitute for P_x, P_y and P_z from equation 4.24, we get

$$P_m \times \left(\frac{x}{d_0}\right)^{\alpha} + P_m \times \left(\frac{y}{d_0}\right)^{\alpha} + P_r < P_m \times \left(\frac{z}{d_0}\right)^{\alpha} \tag{4.26}$$

The energy cost of the reception can be regarded as equivalent to transmitting over a distant d. Thus applying the Log-distance path loss model P_r may be replaced by a virtual distance d given by the formula:

$$P_r = P_m \times \left(\frac{d}{d_0}\right)^{\alpha} \tag{4.27}$$

and finally after replacing it in equation 4.26 and simplifying we get:

$$x^\alpha + y^\alpha + d^\alpha < z^\alpha \tag{4.28}$$

This inequality has to be solved for every communication environment (and thus value of α) separately. I show the solution for the case when $\alpha = 5$. The reasoning is similar for other potential values of α (see section 4.2.2.1). I consider the applications where the network is very dense so the source, the relay and the base station are aligned (see section 4.2.1.2). So, I can rewrite inequality 4.28 as:

$$x^5 + (z-x)^5 + d^5 < z^5 \tag{4.29}$$

After simplification and rearrangement of terms we get:

$$5zx^4 - 10z^2x^3 + 10z^3x^2 - 5z^4x + d^5 < 0 \tag{4.30}$$

I am looking for the locations of the relay for which this inequality is true. So I solve it for x. I apply Ferrari's method [6] for quartic equations: $A = 1/2z^2$, $B = 0$, $C = -3/16z^4 + d^5/(5z)$ and inequality 4.30 is equivalent to

$$u^4 + Au^2 + C < 0 \tag{4.31}$$

where $x = u + z/2$. For this inequality $\Delta = z^4 - 4/5 \times d^5/z$ so, possible solutions for u are:

$$u_1^2 = \frac{-1/2z^2 + \sqrt{\Delta}}{2} \; ; \; u_2^2 = \frac{-1/2z^2 - \sqrt{\Delta}}{2} \tag{4.32}$$

α	source-relay distance when two-hop routing is energy optimal	reception energy condition
2	$x\epsilon\left(\frac{z}{2}-\frac{\sqrt{z^2-2d^2}}{2};\frac{z}{2}+\frac{\sqrt{z^2-2d^2}}{2}\right)$	$z > d\sqrt{2}$
3	$x\epsilon\left(\frac{z}{2}-\frac{\sqrt{3z(3z^3-4d^3)}}{6z};\frac{z}{2}+\frac{\sqrt{3z(3z^3-4d^3)}}{6z}\right)$	$z > d\sqrt[3]{\frac{4}{3}}$
4	$x\epsilon\left(\frac{z}{2}-\sqrt{\frac{-3/2\times z^2+\sqrt{4z^4-2d^4}}{2}};\frac{z}{2}-\sqrt{\frac{-3/2\times z^2+\sqrt{4z^4-2d^4}}{2}}\right)$	$z > d\sqrt[4]{\frac{8}{7}}$
5	$x\epsilon\left(\frac{z}{2}-\sqrt{\frac{-1/2\times z^2+\sqrt{z^4-4/5\times d^5/z}}{2}};\frac{z}{2}+\sqrt{\frac{-1/2\times z^2+\sqrt{z^4-4/5\times d^5/z}}{2}}\right)$	$z > d\sqrt[5]{\frac{16}{15}}$

Table 4.2: *Multi-hop energy efficiency advantage for different values of α.*

The former will never provide real values for x. Substituting for u_2 we can write:

$$x_1 = \frac{z}{2} + \sqrt{\frac{-1/2z^2 + \sqrt{\Delta}}{2}}\ ;\ x_2 = \frac{z}{2} - \sqrt{\frac{-1/2z^2 + \sqrt{\Delta}}{2}}$$

There are real solutions when $-1/2z^2 + \sqrt{\Delta} > 0$ and $z^4 - 4/5 \times d^5/z > 0$ which results in

$$z > d\sqrt[5]{\frac{16}{15}} \tag{4.33}$$

So for the given values of x, y, z and d inequality 4.30 is never satisfied when the reception cost is above certain value ($d > z\sqrt[5]{\frac{15}{16}}$). However, in the other case inequality 4.29 is satisfied when $x_2 < x < x_1$.

Table 4.2 shows the solution for the range of values of α likely to occur (see section 4.2.2.1). Because the value of d is never equal to zero (the reception energy is not negligible), there are intervals of distance x where multi-hop communication is disadvantageous. These ranges are symmetrical about the midpoint of the total transmission distance. So in order to make two-hop routing more energy efficient than a single-hop, the relay should not be too close to either the source or the base station. In the table I also specify the conditions when multi-hop communication is potentially advantageous. If the source-destination distance is too short, the energy saved by relaying does not compensate for the additional reception cost.

In Figure 4.12 the minimum hop transmission distance which guarantees multi-

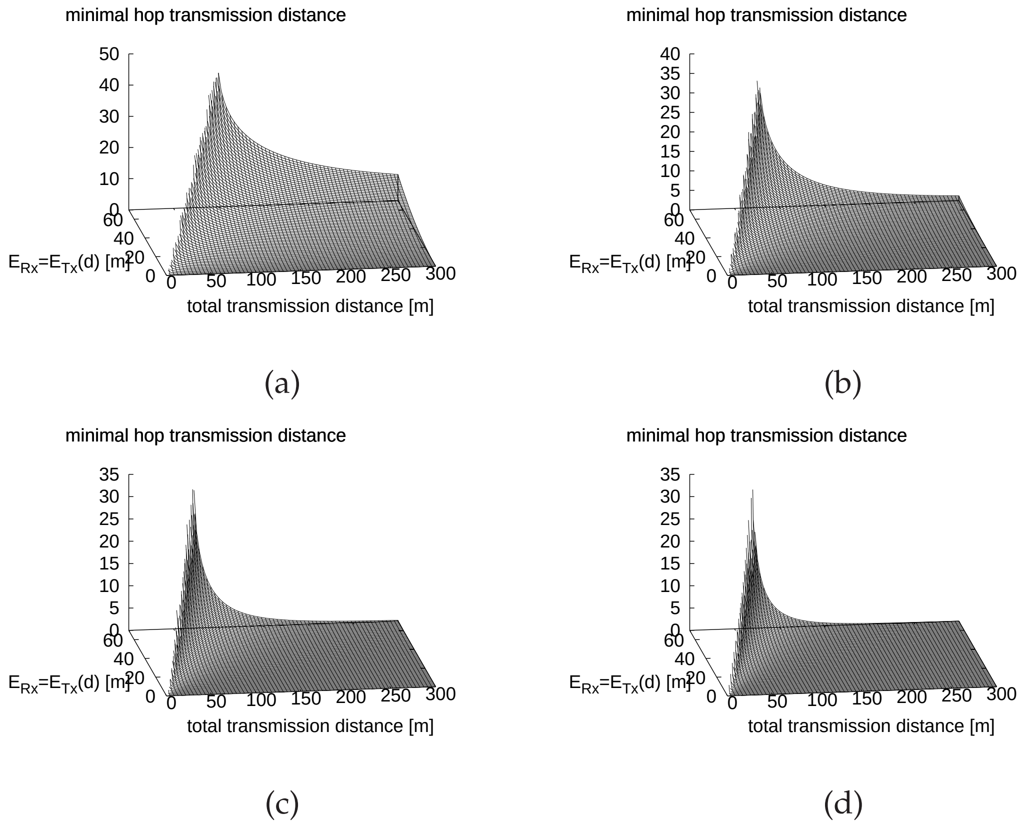


Figure 4.12: *Minimum hop distance in a two-hop transmission for different values of path loss: (a): α=2,(b): α=3, (c): α=4, (d): α=5.*

hop routing energy efficiency in a two-hop scheme is plotted. The conclusion depends on the total communication range and reception cost. So for example, if we consider the case of $\alpha = 2$, a distance from source to base station of 150 meters, and the reception energy corresponding to the 70 meters transmission range ($E_{Rx} = E_{Tx}(70)$), then the two-hop routing is more advantageous than direct communication if the relay is at least 19 meters from both the base station and the source.

The curves have some similarities. There exists a range of transmission distances where direct communication should occur. If the reception cost drops, this interval becomes shorter. For higher values of α the two-hop routing scheme becomes advantageous for smaller transmission distances.

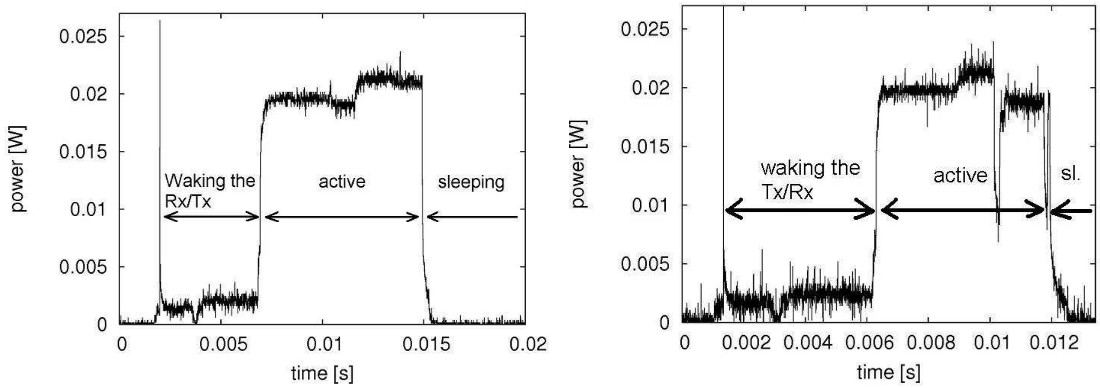


Figure 4.13: *Measured total power consumption when the node is receiving (a) and sending (b) data.*

4.2.2.3 Total energy drained from the node

I next find the relationship between the energy being drained from the batteries and the corresponding radiated energy. I measured the power consumption when the node is transmitting and receiving data (see Figure 4.13). The ratio between the energy use in the sleeping mode and when communicating is significant and can differ by up to a hundred times. This is why we need only consider the energy drained by the transceiver and neglect other sensor node subsystems. Figure 4.10 summarises the relationship between power consumed and that radiated by different transceivers. Since the relationship between system energy consumption and transmitted signal power is monotonic, the conclusions in table 4.2 apply when minimising overall energy dissipation, and not just radiated power.

4.2.2.4 Two-hop optimal relay location

Having determined the conditions which make multi-hop routing advantageous over direct transmission (using the MER metric), I investigate the optimal location of the relay to maximise the energy saved by two-hop routing. I start the analysis by expressing the difference between the radiated power in a two-hop system and one using direct communications. With the Log-distance path loss model and Log-normal shadowing (described in section 2.4.1) we can represent the power received at a distance d from the source by combining equations 2.2

and 2.5 as:

$$P_r(d) = \beta \times \frac{P_t}{d^\alpha} \tag{4.34}$$

where β is a constant specified in the model definition and P_t is the RF output power. So the power saved by the two-hop transmission is:

$$P_{saved} = P_z - (P_x + P_y + P_r) \tag{4.35}$$

with P_z, P_x and P_y representing transmission power over distances z, x, y (see Figure 4.11) and P_r denoting the reception cost. If I assume nodes send data using the minimum energy to reach the destination and P_r is equivalent to communication over distance t (see section 4.2.2.2) equation 4.35 becomes:

$$P_{saved} = \frac{P_m}{\beta} \left[z^\alpha - (x^\alpha + y^\alpha + t^\alpha)\right] \tag{4.36}$$

In order to find the maximum of P_{saved} I consider equation 4.36 for a sample value of $\alpha = 5$. For other cases the reasoning and conclusion are similar. As stated before, the source, the relay and the base station are aligned so that equation 4.36 becomes

$$P_{saved} = \frac{P_m}{\beta} \left[z^5 - (x^5 + (z - x)^5 + t^5)\right] \tag{4.37}$$

After simplification and rearrangement of the terms we get

$$P_{saved} = \frac{P_m}{\beta} \left[5z^4x - 10z^3x^2 + 10z^2x^3 - 5zx^4 - t^5\right] \tag{4.38}$$

I want to find when this function reaches a maximum

$$\frac{\partial P_{saved}}{\partial x} = \frac{P_m}{\beta} \left[5z^4 - 20z^3x + 30z^2x^2 - 20zx^3\right] \tag{4.39}$$

Making a substitution $n = x/z$ equation 4.39 becomes

$$\frac{\partial P_{saved}}{\partial x} = \frac{P_m}{\beta}\left[5z^4 - 20z^4 n + 30z^4 n^2 - 20z^4 n^3\right] \tag{4.40}$$

$$\frac{\partial P_{saved}}{\partial x} = \frac{P_m}{\beta}5z^4\left[1 - 4n + 6n^2 - 4n^3\right] \tag{4.41}$$

$$\frac{\partial P_{saved}}{\partial x} = \frac{P_m}{\beta}5z^4\left[\left(n - \frac{1}{2}\right)\left(-4n^2 + 4n - 2\right)\right] \tag{4.42}$$

$\partial P_{saved}/\partial x = 0$ only for $n = 1/2$ and this derivative changes the value from positive to negative so the function P_{saved} achieves the maximum value for $n = 1/2$. So when the relay is half way between the source and the base station, the power saved by the two-hop routing is maximised. This result may readily be generalised to apply for any positive value of α. It may be concluded that when two-hop communications is more advantageous than direct transmission, the optimal relay location is equidistant from the source and the base station.

4.2.3 Experimental results

In the previous section the analysis is restricted to an ideal chain topology, where nodes are perfectly aligned and propagation noise is neglected. In real scenarios, randomness of the signal can have a significant impact on the received power, and thus the energy efficiency of the communication scheme. In this section, I document experiments performed with a sensor network to verify the conclusions obtained by theoretical analysis.

The network is composed of 10 nodes which are aligned to simulate a communication chain in a dense network. I employed a commonly used sensor network platform: Tmote Sky sensor node [122] and the networking stack as implemented in TinyOS [64]. Each Tmote Sky node has a 2.4GHz, 250kbps IEEE 802.15.4 Chipcon wireless transceiver. Because of its power consumption properties (see Figure 4.10: the maximum transmission power equals 0 dBm) I extrapolate the results for a 20 dBm (which is the maximum allowed transmission power by the IEEE 802.15.4 standard) transmitter in the following way. I place the source and the base station at the maximum possible communication distance. In between, I place a line of potential equidistant relays. I obtain first the path loss coefficient from measurements of received power at every relay (using a linear regression method I obtained $\alpha = 2.6$ and $P_0 = -57$ dBm, see equation 4.23) when the radiated power is 0 dBm. Then I assume that the RF output is 20 dBm. This operation shifts the curve of received power versus distance from the transmitter up by 20 dBm (see figure 4.14). From the measured signal strength at each relay and base station I calculate their extrapolated distance to the source. From figure 4.14 we can deduce that relays which are closer to the transmitter are moved away from it by a longer distance than remote relays. Thus node 1 in figure 4.16 (located at 10% of the source-base station distance) is shifted to 19% of that distance, whereas node 8 remains at about 90% of this distance. In the next step I analyse the energy efficiency of both schemes: direct and two-hop transmission.

To do that I measure the minimal power radiation to reach the base station from each node location and also to send data from every relay to the base station. From this value I calculate the total power consumption by a linear approximation for the XBee-PRO module (see Figure 4.10). The results are shown in Figure 4.15.

From that I can estimate the total energy consumption of direct transmission and communication through each relay. These values are plotted in Figure 4.16, where the node with id=9 is the base station. The curve should in theory be symmetrical with respect to the line x=0.5. The divergence from that stems from the fluctuations of the signal strength. For communication with the base station, node 4 requires a significant RF output in comparison to nodes 3 and 5. We can also observe that there are relay locations for which direct communication with the base

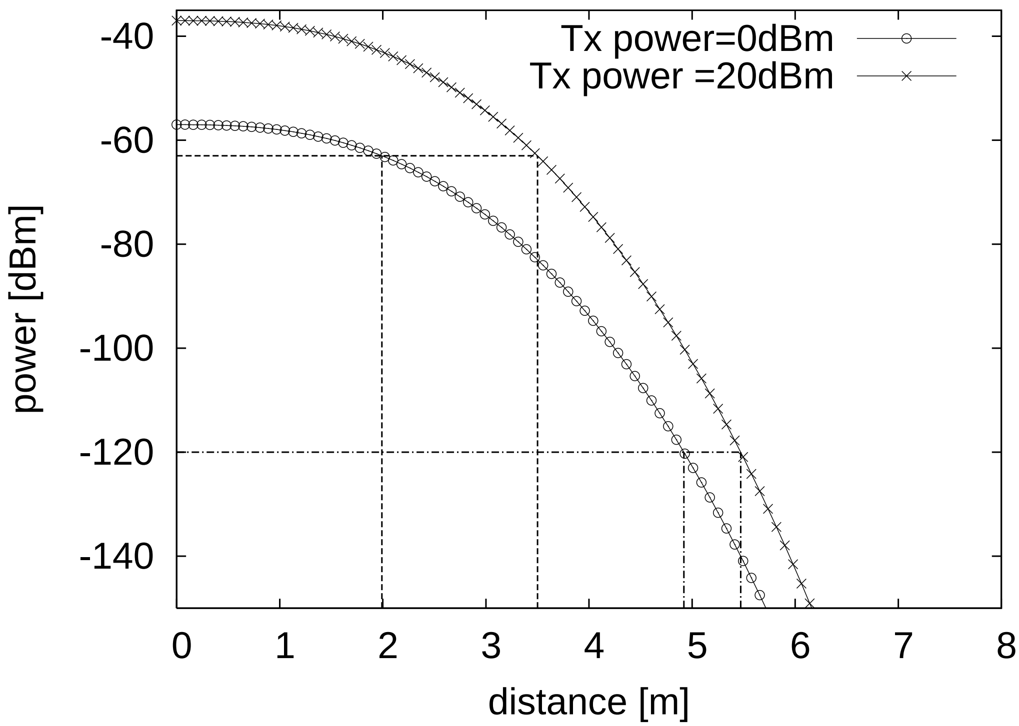
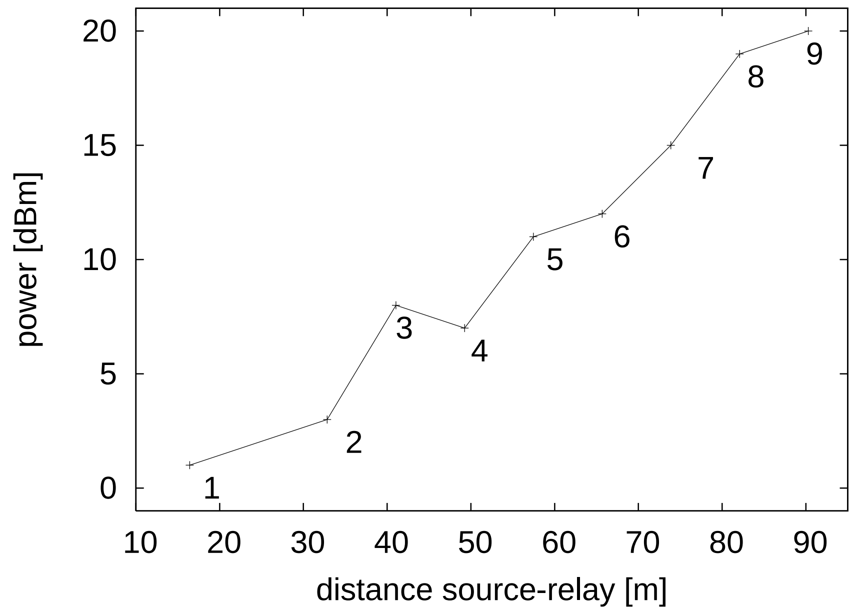


Figure 4.14: *Example of how the extrapolation used for experiments changes distances between nodes. Nodes that are close to the transmitter are moved by a larger distance than remote sensor nodes.*

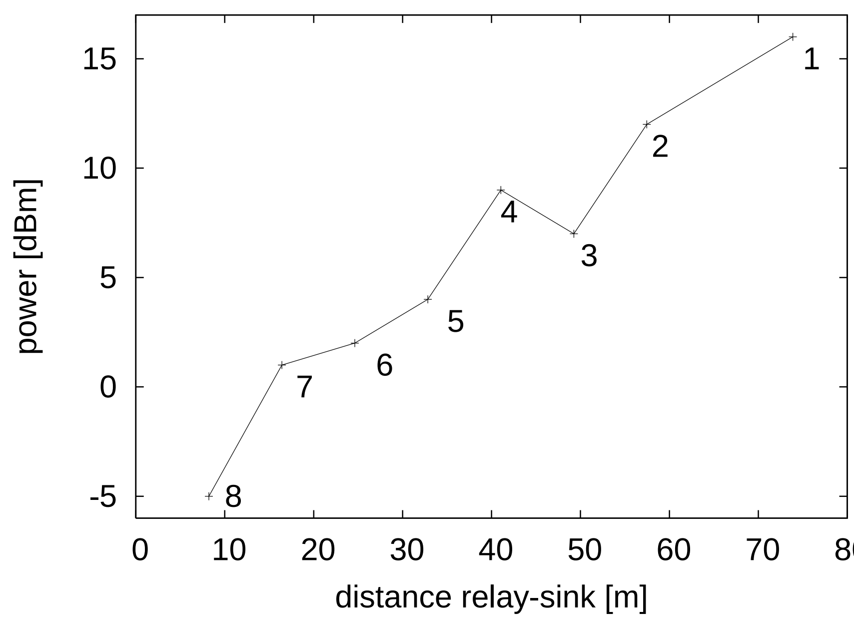
station is more advantageous (*e.g.* node 8). From the experiments I conclude that direct communication is more energy efficient if the hop-distance is below 12m. However the theoretical value for the given topology and environment equals about 13.5m.

4.2.4 Hop-distance influence on energy efficiency of a route: summary

I have analysed the problem of hopping distance strategy in WSNs. I have shown when multi-hopping is more energy efficient in comparison to direct transmission. I also determined the optimal location of the relay for the two-hop communication scheme. These inferences were verified experimentally on sensor networking hardware. The results obtained may be used to optimise routing protocols for WSNs. If location information is available to the network layer, then the



(a)



(b)

Figure 4.15: *Minimum transmit power required for: (a) source communicating with relays (numbers correspond to the node ID, 9 is the base station); (b) relays communicating with the base station.*

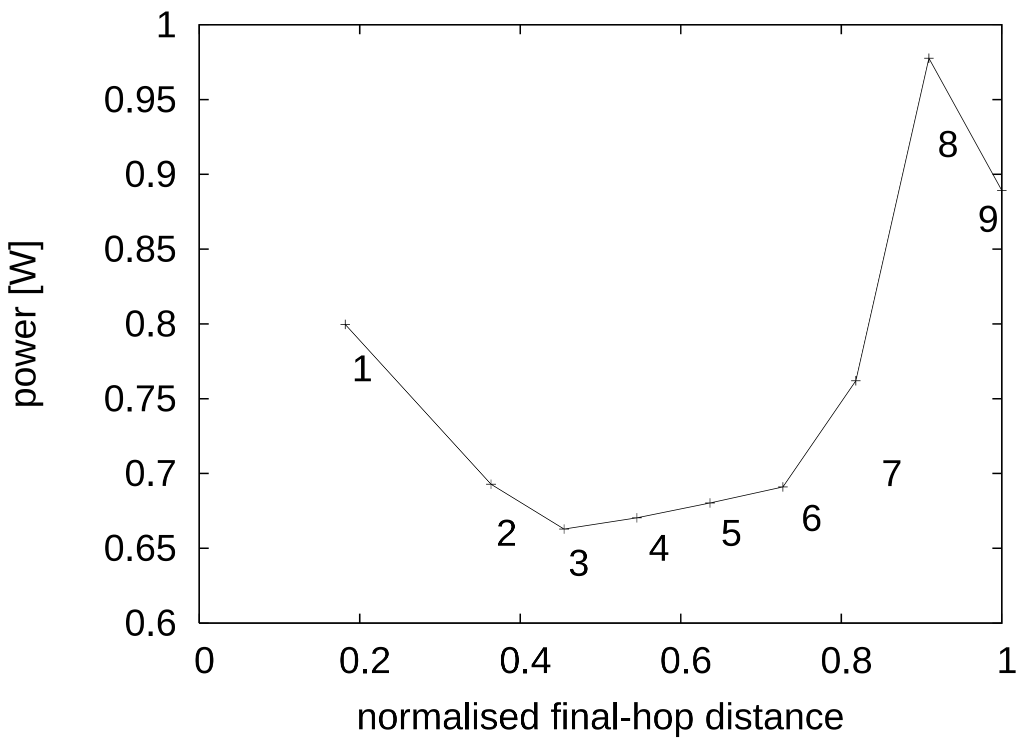


Figure 4.16: *Total power consumption of the one-hop communication with different relays.*

routing protocol can determine the optimal relay node for packets.

4.3 Transition region study

Experimental studies on WSNs have revealed the existence of three distinct reception regions. These regions can be classified as connected, transition (described in section 2.4.2), and disconnected. Their location and size may have a significant impact on the performance of communication protocols, and particularly of routing. Many authors reported that the routing decisions would be different if based on a model that takes into account the existence of unreliable links [85, 161]. Hence, if we can estimate where such links may occur,we can avoid using those links for packet routing. Zuniga *et al.* [163] determined the underlying causes of transition region using analytical techniques from communications theory (see section 2.4.2.3) along with practical results for verification. The proposed solution was verified for a radio architecture that employs NRZ and non-coherent FSK and cannot be directly applied to other communication schemes. In this section I describe how the transition region can be estimated for an O-QPSK architecture and its experimental confirmation.

4.3.1 Inaccuracy of existing transition region model for 2.4GHz O-QPSK architecture

The transition region was indepth studied by Zuniga *et al.* [163]. They developed a model of the transition region using analytical techniques from communications theory and provided empirical results confirming the theoretical analysis. These latter experiments were carried out using a radio architecture that employs NRZ and non-coherent FSK. Zuniga *et al.* made brief remarks about some preliminary empirical evaluations using the Chipcon CC2420 (which uses O-QPSK modulation) transmitting at a nominal output power of -10 dBm [96]. They observed a significant extent of transition region although no specific values were given. However I observed that my empirical results do not exactly follow the theoretical prediction of the transition region given by Zuniga's model. Since their method is not applicable to O-QPSK modulation scheme, it is not surprising that their predictions are at variance with empirical results found below, and that a new model to cover such schemes is required.

4.3.1.1 Theoretical delineation of transition region

The PRR for a transceiver using DSSS with O-QPSK modulation, is related to the BER as follows (see equation B.2):

$$\text{PRR} = (1 - p_e)^{8f} \tag{4.43}$$

where p_e is the BER and f is the frame length of the packet (in bytes). This equation is not directly applicable to O-QPSK with DSSS schemes. However, this is a close approximation of the true PRR expression used also by other researchers [73].

The value for p_e was taken from the IEEE 802.15.4 standard [3]:

$$p_e = \frac{8}{15} \times \frac{1}{16} \times \sum_{k=2}^{16} -1^k \binom{16}{k} e^{20 \times \mathrm{SINR} \times (\frac{1}{k} - 1)} \quad (4.44)$$

$$P_r(d) \propto \frac{P_t}{d^\alpha} \quad (4.45)$$

Using this and equation 4.43 rearranged as:

$$p_e = 1 - \mathrm{PRR}^{\frac{1}{8f}} \quad (4.46)$$

the values for γ_U and γ_L (described in appendix B) were calculated as $-.3$ and -2.3 respectively. These were then used in equations B.16, B.17 and B.18 to calculate the extent of the transition region. To obtain that result I determined empirically (with experiments described in the next section) the channel parameters (decay factor η and standard deviation σ) of an indoor and an outdoor environment. These can be seen in table 4.3 and result in $d_s = 11.5$ m, $d_e = 103.1$ m and $\Gamma = 8.0$ for the indoor environment with $d_s = 4.7$ m, $d_e = 15.1$ m and $\Gamma = 2.2$ for the outdoor environment. The method for empirical estimation of the transition region in these environments along with the obtained results are described in the following section.

Environment	η	σ	$PL(d_0)$
Outdoors	3.6	4.11	49.7 dB
Indoors	2.6	5.8	50.7 dB

Table 4.3: *Channel Parameters. The reference value d_0 is 1m for the outdoor environment and 2m for the indoor environment.*

4.3.1.2 Empirical verification of theoretical analysis

In order to determine the PRR, twenty Tmote Sky sensor nodes [122] were evenly spaced in a linear setup for both an indoor and outdoor environment. These sensor nodes are equipped with a Chipcon CC2420 transceiver [23] which is a spread spectrum based device. The CC2420 operates at a frequency of 2.4GHz and I chose to use channel 26 in my experiments as this lies outside the operating frequency range of IEEE 802.11 b/g for example and would therefore help to reduce interference (see Figure 4.21). The transmit power was set to -15 dBm and each node sent 6000 packets in turn. The remaining nodes (all in receive mode) determined the number of packets correctly received from the transmitter along with the average RSSI. Once all twenty nodes finished transmitting, the data was downloaded to a laptop for analysis. The experiment was repeated five times.

The indoor environment was a corridor within my university and an open field was used for the outdoor tests. Figures 4.17 and 4.19 show plots of the PRR for both environments whereas Figures 4.18 and 4.20 show plots of the RSSI. From these plots it was determined that $d_s = 4$ m and $d_e = 32$ m for the indoor environment, resulting in $\Gamma = 7$. Comparing this with the results of Zuniga *et al.* where $d_s = 6.9$ m, $d_e = 25.9$ m and $\Gamma = 2.8$ it would appear straight away that the transition region is not reduced by using a spread spectrum architecture. The outdoor environment is quite different where it can be seen that $d_s = 4$ m, $d_e = 11$ m and $\Gamma = 1.8$. Again comparing these with [163] where $d_s = 3.2$ m, $d_e = 8.6$ m and $\Gamma = 1.7$ we see that the transition regions are very similar in this case. This is to be expected as the effects of the spread spectrum system on the transitional region are less noticeable due to reduced levels of multi-path. The fact that the transition region is a lot larger for the indoor environment is in agreement with the results of [163] and is caused by a smaller decay factor and larger standard deviation.

It can be seen that the empirical and analytical values of σ agree quite closely for both environments. However, for the indoor environment, d_s and d_e disagree

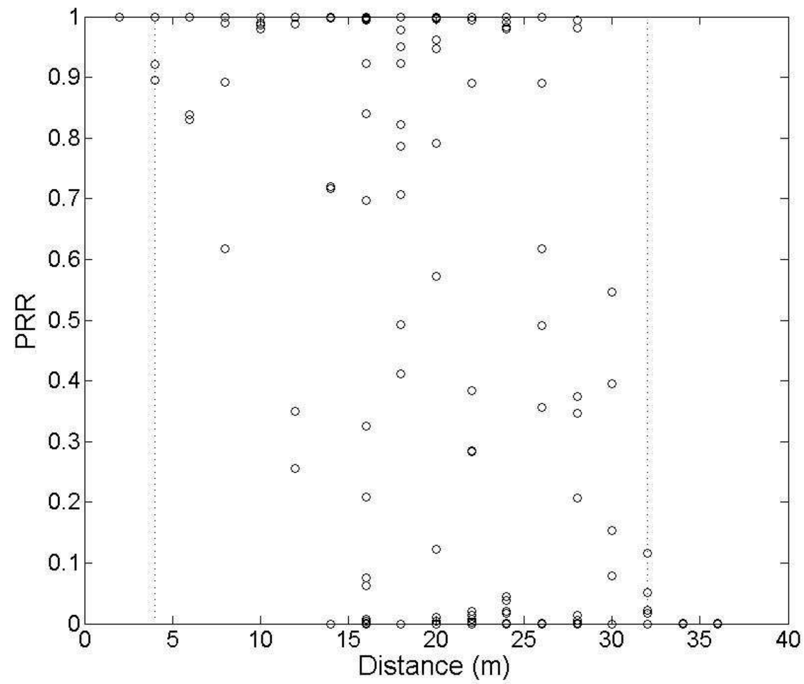


Figure 4.17: *PRR indoors with each sensor node separated by a distance of 2m with the nodes placed on the ground. The transmit power level is set to -15 dBm.*

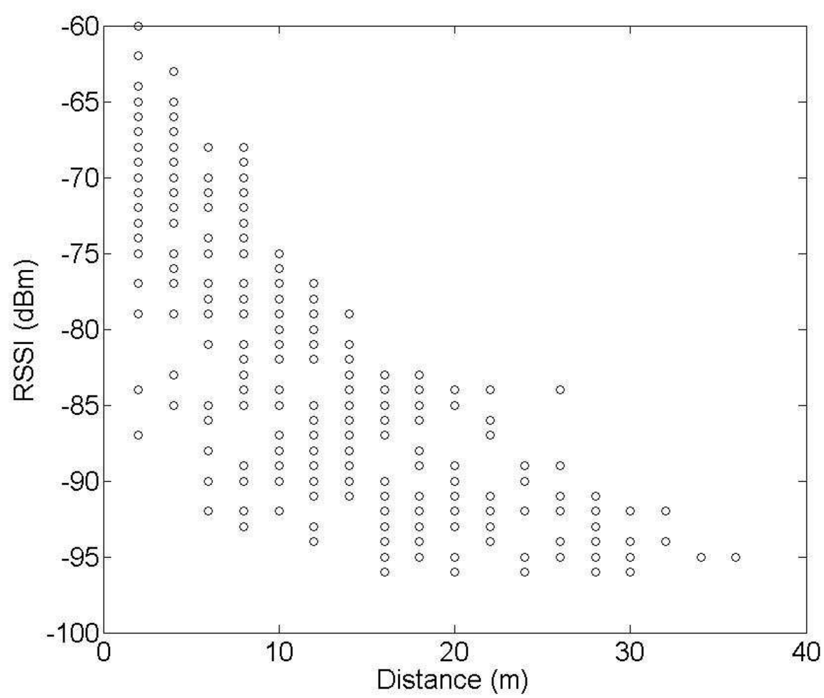


Figure 4.18: *RSSI indoors with each sensor node separated by a distance of 2m with the nodes placed on the ground. The transmit power level is set to -15 dBm. η is approximately 2.6 and σ is approximately 5.8.*

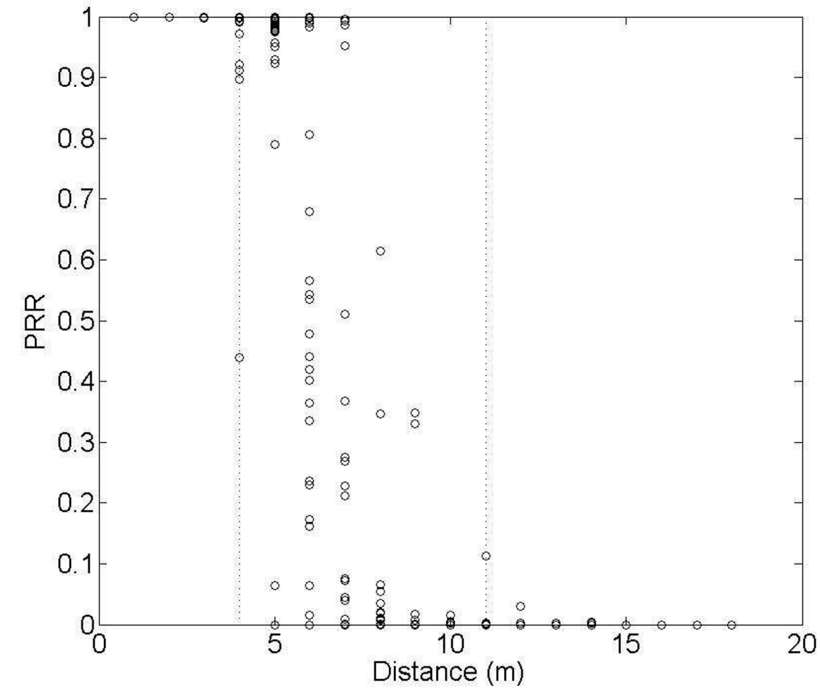


Figure 4.19: *PRR outdoors with each sensor node separated by a distance of 1m. The nodes were placed on small cardboard boxes about 5cm in height. The transmit power level is set to -15 dBm.*

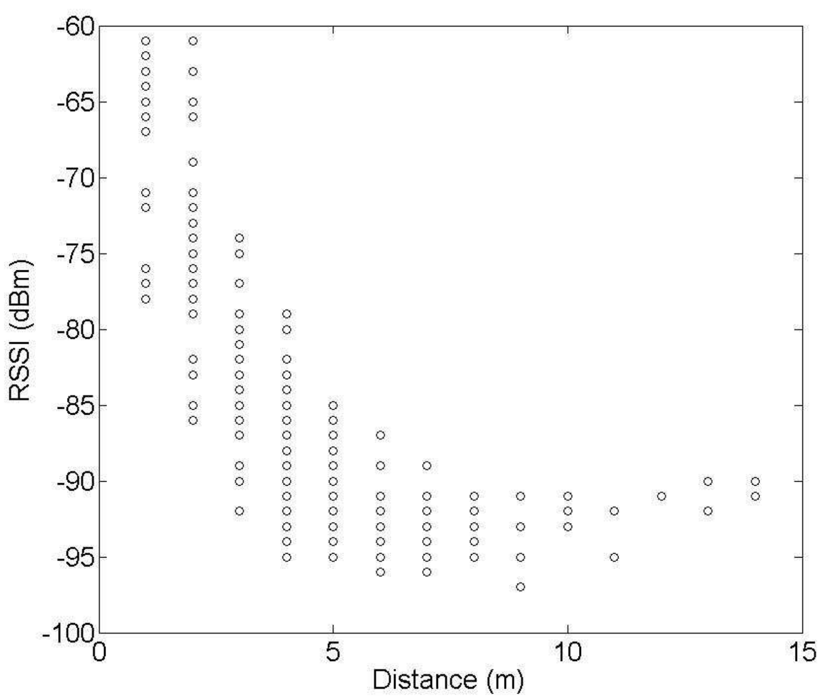


Figure 4.20: *RSSI outdoors with each sensor node separated by a distance of 1m. The nodes were placed on small cardboard boxes about 5cm in height. The transmit power level is set to -15 dBm. η is approximately 3.6 and σ is approximately 4.11.*

quite strongly. The same is true for the outdoor environment although to a lesser degree.

4.3.2 A new method of transition region estimation

In my opinion the difference between the empirical and theoretical determinations of the transition region is due to the estimation of SNR on the basis of the RSSI value which is proposed in [163] (see section 2.4.2.3). Although the method for doing this is not given by the authors, the only possible approach is to use the following equation:

$$\mathrm{SNR} \approx 10\log\left(\frac{\mathrm{RSSI} - P_n}{P_n}\right) \tag{4.47}$$

where RSSI and P_n represent the received signal strength and noise floor respectively, both of which are in Watts. Equation 4.47 is only an estimate as it neglects interference from other sources that may be included in the measured RSSI value. However, I minimised this interference by using channel 26 for experiments, as it lies outside the range of 802.11 b/g (see Figure 4.21). Despite this precaution the difference between empirical and analytical values is substantial. The discrepancy is smaller for the outdoor environment. This suggests that the estimation of SNR using the value of RSSI (equation 4.47) is incorrect because this neglects the interference caused by multipath fading. In the indoor environment this phenomenon is more likely to be observed because of many potential obstacles between communicating nodes. The outdoor environment is an open space where eventual sporadic obstacles have only a small impact on propagation of the signal.

Hence I propose to overcome this problem by empirically estimating the BER in order to identify the transition region. So instead of applying the general BER equation (equation 4.44) for every environment I recommend use of the BER curve appropriate to each environment. The BER should be a function only of SNR, and thus should be independent of other channel characteristics. How-

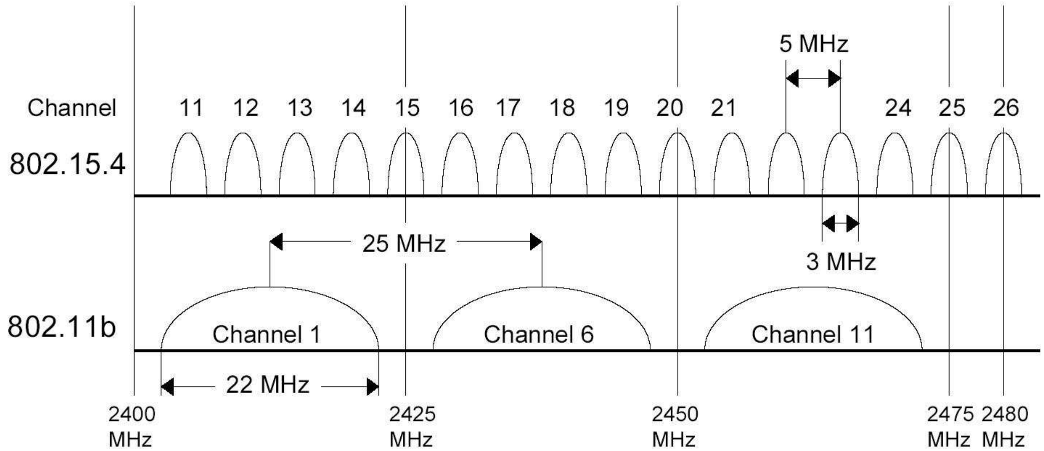


Figure 4.21: *IEEE 802.11b and IEEE 802.15.4 channel selection.*

ever, SNR is difficult to measure empirically using typical commercially available WSN transceiver designs. These typically support the measurement of RSSI, from which we can estimate the SNR. However, RSSI measurement is channel dependent, the resulting estimate of SNR will share this dependence, a variation which would not be present in the true SNR value.

4.3.3 Experimental results

To confirm my hypothesis that the RSSI does not incorporate the multipath effect and thus that the delineation of transition region is incorrect with the method described in section 4.3.1.1 I carried out the tests of BER for different environments and used the values obtained to estimate the transition region extent.

4.3.3.1 Bit Error Rate (BER) tests

In order to carry out the BER test I used a pair of Tmote Sky sensor nodes. The CC2420 transceiver can be configured to run in a number of different modes one of which is an unbuffered serial mode. In this mode the transmitter sends a synchronisation sequence to the receiver followed by a constant stream of data bits at 250Kbps. In my setup, upon receiving a Start of Frame Delimiter (SFD), the CC2420 forwarded the decoded bits directly to the microcontroller (without buffering) which then compared them to reference data and updated the BER for a corresponding RSSI. The RSSI value itself was read directly from one of the CC2420 registers.

A major problem was encountered in serial mode because the MSP430 operates at or below 8MHz[1] while the CC2420 operates at 16MHz. The CC2420 transmits at a data rate of 250kbps. However due to the slower clock speed of the MSP430 it is not fast enough to process the incoming bits received by the CC2420. To overcome this problem a simple frequency divider circuit was placed between the transceiver and the microcontroller. This allowed the MSP430 to collect every 32nd bit in the sequence. The divider itself was used to divide the synchronisation clock which occurs on the FIFOP pin of the CC2420.

Once the frequency divider circuit was in place, a random sequence of a million bits was generated at the beginning of each test. These bits were then transmitted to the receiver n times. The RSSI was sampled at the receiving node once every 32 bits and a histogram of the number of bits received at a particular RSSI obtained. Experiments were run at two different locations (in a narrow corridor and a spacious laboratory) and about 10 million bits were collected for each RSSI value. An estimation of the noise floor was also obtained by sampling the RSSI register with no sensor nodes transmitting. This allowed me to estimate the SNR using equation 4.47.

The noise floor can be determined analytically using the equation:

$$P_n = FkT_0B \tag{4.48}$$

where F is the noise figure of the radio, T_0 is the ambient temperature and B is the noise equivalent bandwidth [126]. The value of F can be taken to be about 11 to 12 dB but the figure for B could not be found. As an estimate, the bandwidth of the channel was used (defined to be 3MHz in the standard).

[1]The clock speed is determined by the presence or absence of the "rosc" resistor attached to port 25 of the microcontroller - see [132].

Using these data an approximate value for the noise floor is obtained:

$$P_n(dB) = 12 - 198.6 + 10\log(300) + 10\log(3 \times 10^6) \approx -97\text{dBm} \tag{4.49}$$

This corresponded quite closely with the measured values which were between -94 and -98 dBm.

The results of the BER tests can be seen in Figure 4.22. Included here is a plot of equation 4.44 for comparative purposes. The initial tests were performed in the corridor and bits were sent at 0 dBm so the nodes had to be placed about 20m apart in order to capture any errors. I observed that the empirical results were considerably different from the analytical values given in the standard. This discrepancy is mainly due to multipath effects. Tests performed in the laboratory where two nodes were placed on a table with no obstacles in the vicinity support that hypothesis. The transmission power was set to -25 dBm and errors were observed at very close proximity (around 1m). As can be seen from Figure 4.22 the empirical values of the BER obtained in this case are a lot closer to the theoretical estimation although a discrepancy still exists.

The differences in both cases are mainly due to the multipath effects described previously (in section 4.3.2) which are not reflected in the measurement of RSSI. Other phenomena may also be influential.

- The theoretical formula provided by the IEEE standard may not be completely suitable for the CC2420 for the following reasons. The modulation scheme used by the IEEE 802.15.4 (and hence the CC2420) is similar to Minimum Shift Keying (MSK) (*i.e.* O-QPSK with half sine pulse shaping). O-QPSK has the same theoretical bit error performance as BPSK and QPSK (assuming coherent detection). The same is true for MSK detected using a matched filter to recover each of the quadrature components independently [138]. Equation 4.44 is clearly quite different to those for BPSK and

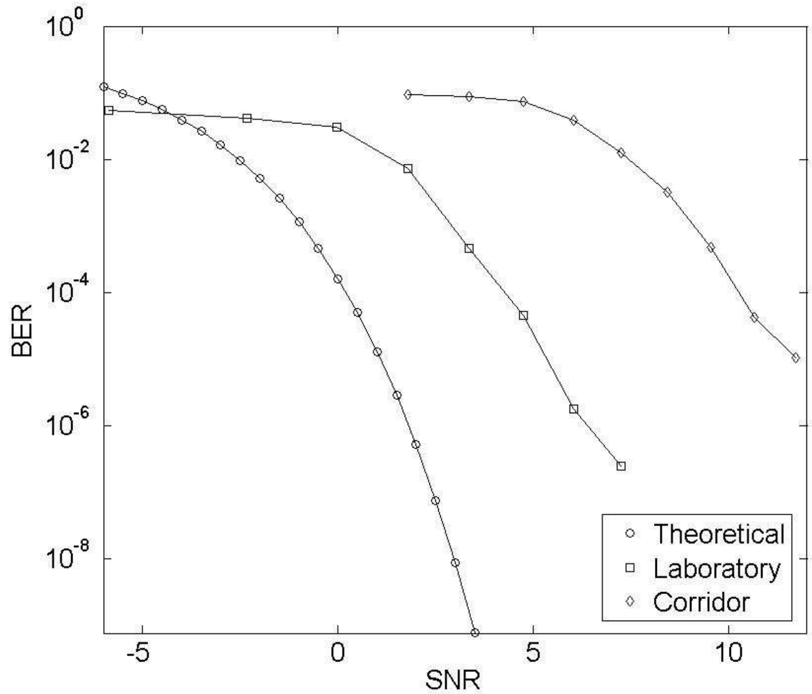


Figure 4.22: *Plot of BER given in IEEE 802.15.4 standard [3] compared with the empirical measurements using the CC2420 transceiver.*

QPSK. This is because MSK can also be viewed as a special form of CPFSK where the deviation index is exactly equal to $\frac{1}{2}$ [125]. However, the error probability for CPFSK also depends on whether the detector is coherent or non-coherent and whether it uses symbol-by-symbol detection or sequence estimation [50]. The CC2420 uses coherent detection with sixteen 32 chip pseudo-orthogonal codes. For these reasons it was thought possible that equation 4.44 might not be a perfect model for my system as the CC2420 is just one of a number of possible implementations of the standard's specifications.

- The transceiver does not estimate the RSSI value for every bit but calculates an average over 32 consecutive bits. Even though the standard deviation of the RSSI values over this period is quite small, the average value read from the CC2420 register may differ from a single bit measurement by 1 or 2 dBm.

- The efficiency of the synchronisation performed at the beginning of the test may also have had an influence on the shape of the BER curve. The RSSI value may deteriorate during the synchronisation time and even if the signal strength improves, the consequences of this may still be visible in the BER.

- The approximation of the SNR using equation 4.47 is almost certain to have an influence on the BER.

All these factors combined contribute to a difference between the theoretical and empirical BER estimation. As a side point, the results I obtained would suggest that the RSSI is generally not a very good estimator of the PRR because, as I have observed, for the same RSSI value in two different environments the BER was quite different. Also other researchers have come to a similar conclusion [37, 38, 122, 160]. This would seem to contradict the results in [143]. However it is possible that under reduced levels of interference RSSI may in fact be quite a good estimator of the link quality. This can be seen in Figure 4.22 where the BER curves move closer to the theoretical value as the interference levels are reduced[2].

For the experiments carried out in section 4.3.1.2 a frame size of 39 bytes (28 bytes payload and 11 bytes header) was used giving $8f = 312$. Using this value in equation 4.46 and the empirical results obtained from the BER measurements, the values of γ_U and γ_L can be determined. Table 4.4 shows these results along with those obtained for the theoretical BER equation in section 4.3.1.2. Interestingly, the difference between γ_U and γ_L is approximately the same in each case. Considering this and inspecting Figure 4.22, it is clear that all of the curves have a very similar shape, only differing by a reasonably constant offset.

One possible explanation for this is again due to equation 4.47 which uses the RSSI to calculate the SNR. Although it is not stated in the CC2420 data sheet how the RSSI is determined, it is likely that the value measured consists of the

[2]The most likely cause of this interference is multi-path fading.

	γ_U	γ_L
Theoretical	-.3	-2.3
Laboratory	3.57	1.77
Corridor	9.7	7.7

Table 4.4: *SNR values.*

power of the received signal plus any interference. In the corridor there would have been more interference than in the laboratory due to the particular setup used. The SNR calculated would therefore have been an overestimation of the actual SNR which would explain the curve moving towards higher SNR values in Figure 4.22.

Table 4.5 shows the overall results obtained for the transitional region. The values of Γ obtained using the three BER curves of Figure 4.22 show similar results in each environment. This is easily explained by referring to equation 4.47 and noting that the difference between γ_U and γ_L is approximately the same for each curve. As can be seen for both the indoor and outdoor environments, the results agree quite nicely with the empirically obtained values.

	Indoors			Outdoors		
	d_s	d_e	Γ	d_s	d_e	Γ
Theoretical	11.5	103.1	8.0	4.7	15.1	2.2
Laboratory	8.2	72.3	7.8	3.7	11.6	2.2
Corridor	4.8	43.1	8.0	2.5	8.0	2.2
Empirical (PRR)	4.0	32.0	7	4.0	11.0	1.8

Table 4.5: *Transitional Region Parameters. Empirical (PRR) are the results obtained in section 4.3.1.2*

A discrepancy occurs however when comparing d_s and d_e in some cases with the empirical values. For example, the values obtained using the theoretical BER curve are in strong disagreement with the empirical results for the indoor environment and, as mentioned, to a lesser degree for the outdoor environment. This is to be expected as only one of the curves should correspond to the actual value. The values obtained for the indoor environment using the corridor BER agree quite closely with the actual values. In the outdoor case the values obtained using the results from the laboratory more closely agree. Again this is likely due to the approximation error introduced by equation 4.47. As RSSI values were used to calculate both the analytical and empirical results it would seem plausible that similar environments would produce similar results. This claim is supported by the observation that if the curve of Figure 4.20 is shifted down by approximately 4 dB (the same difference between the theoretical and laboratory BER curves of Figure 4.22) and the value of PLd_0 recalculated as $49.7dBm + 4dB = 53.7dBm$, then the transitional region is best approximated by the theoretical curve of Figure 4.22 giving values of $d_s = 3.6$, $d_e = 11.7$ and $\Gamma = 2.2$.

The described method of transition region estimation can be used to optimise communication performance. It can be integrated in the routing protocol in order to eliminate communication over unreliable links. Below I describe such optimisation of routing protocol using a cross-layer approach.

4.3.4 Integration of the method into routing layer

Many studies have revealed numerous disadvantages of communication over lossy links [85, 161]. I propose that every sensor node avoid routing packets over such links by estimating the extent of the transition region and excluding the neighbours which belong to it from the routing table. To do this, the routing protocol needs to obtain the following information from other network stack layers:

- **BER** as a function of the SNR can be estimated by the physical layer, using the RSSI measurements (available from most current WSN sensor nodes) the equation 4.47 to calculate SNR. Then the BER values corresponding to the starting and ending limits of the transition region can be obtained with equation 4.46. Next, from the BER measurements the values of γ_U and γ_L (which are the values of SNR corresponding to the limits of the transition region) can be deduced.

- **Wireless channel parameters** are needed to calculate the numerical values of d_s and d_e (see equation B.16 and B.17). The physical layer can estimate the transmit power (P_t) , the power lost at reference distance d_0 ($PL(d_0)$), the standard deviation of the shadowing component (σ), the noise floor (P_n), and the decay factor (η).

- **Location of the nodes** is required for both, transmitter and receiver, to delineate the transition region and to find out if a neighbour is located in it. Many localisation protocols have been proposed in the past [79]. Once the node knows its location and that of all its neighbours it should make this information available to the routing layer for use in determining the extent of the transition region. This can be represented by the values of d_s and d_e which can be calculated with equations B.16 and B.17 (all required components of the equations having already being obtained).

Finally the routing protocol can exclude the nodes which belong to the transition region from the routing tables. This can be done by comparing the location information of every neighbour with the values of d_s and d_e. If a node is located at a distance between d_s and d_e from a transmitter, then the transmitter should not route the packets through this node.

4.3.5 Transition region study: summary

Communication over unreliable links may cause poor communications performance of WSNs. These links occur when the receiver is located in a so called transitional region. Hence, in the previous sections I proposed a method of determining the extent of this region for an O-QPSK 2.4GHz architecture. This applies a cross-layer information sharing between network stack layers and can be used by a routing protocol to eliminate the communication with neighbours located in the transition region.

4.4 Summary

In this chapter I have investigated different cross-layer methods of optimisation of routing protocols which use location information. Many applications already require location information and thus for a minimal additional cost, by making this information available to the network layer, it is possible to overcome the following problems which I have studied in this chapter:

- First, I study optimisation of intra-cluster topology in terms of energy efficiency. To extend the lifetime of the network I proposed a method which integrates information about node location into the routing layer. It uses the angle between the node, cluster-head and the base station as a criterion to govern routing protocols choice of cluster-head. To show an example of implementation of this model I integrated it with LEACH which results in a novel protocol called Smart Clustering Adapted LEACH (SCALE). The experiments show a significant decrease of the energy dissipation and thus an increase of network lifetime when using the SCALE protocol instead of LEACH.

- I also investigate the influence of the hop distance strategy on the WSN lifetime. As a result of my study I determine when multi-hop routing is more energy efficient than direct transmission to the base station. Also, I define the conditions for which the two-hop strategy is optimal. Experimental evidence is provided to support of these conclusions. The tests showed that the superiority of the multi-hop scheme depends on the source-base station distance and reception cost. They also demonstrated that the two hop strategy is most energy efficient when the relay is at the midpoint of the total transmission radius. My results may be used in existing routing protocols to select optimal relays or to determine whether it is better to send packets directly to the base station or through intermediate nodes.

- Moreover, I explore the extent of transitional region in WSN. Communication with nodes located in this region may lead to a low packet delivery ratio and worsen the performance of a protocol which implements flooding and reverse path forwarding. Thus I present a model for delineating the transition region to avoid communication with sensor nodes located within its boundaries. The experiments proved that my method is more precise than other existing models for O-QPSK 2.4GHz architectures.

The routing optimisation methods presented in this chapter show how location information may be used to improve routing protocols in WSNs. They suggest that to achieve a high energy efficiency of routing in WSNs and to overcome the problem of communication difficulties in the transition region, the protocols must use cross-layer information sharing.

CHAPTER 5

Conclusions

Wireless Sensor Networks are attracting considerable interest and are a very promising technology. Due to the limited storage, energy, and computational resources of WSNs, the routing techniques developed for other types of network are not adequate for them. Although some of these methods can route packets in a WSN, they cannot deliver sufficient energy performance to ensure adequate network lifetime and therefore, ongoing research is required on efficient WSN routing strategies.

Many WSN routing research proposals apply a traditional layered approach to the design of a protocol to comply with the principle of modularity in system development. This layered independent architecture may be unduly restrictive in the WSN context as it can lead to a significant drop of communication performance. The solutions advocated here for overcoming the overhead incurred by traditional layered routing architectures use the so called cross-layer method where communication between nonadjacent layers is enabled. The advantage of these novel methods was tested by simulations and by experiments carried out with WSNs deployed in different environments and various communication scenarios.

5.1 Contributions

In this thesis I proposed two groups of routing protocol optimisations incorporating time information and node location information into the network layer by using a cross-layer design approach. The time synchronisation problem is a standard problem in WSNs: it is required at the interface between the sensor network and an external observer (for scheduling, reporting results, and management of the WSN), among the nodes of the sensor network (for intra-network coordination among different sensor nodes *e.g.* concurrency control), and at the interface between the sensor network and the observed physical world (when for example data fusion may extract higher-level information about the observed object). Besides these needs I also identified that for some protocols (*e.g.* LEACH) the autonomous operation of the network stack layers can lead to a significant drop of communication performance. To prevent these failures and also to optimise the energy efficiency of the routing protocol I propose two methods which integrate time synchronisation into the routing scheme. They are complementary and are designed for different types of network architectures.

The first method, called Cross Layer Efficient Architecture for Routing (CLEAR) is designed for cluster-based networks. It efficiently distributes schedules to subordinate nodes and by minimising their awake time it improves the network lifetime. CLEAR also enables packet delivery guarantees because the packet reception rate can be very low in WSNs due to the harsh transmission environment and the limited computational resources of nodes.

The second method, called Routing Integrated Synchronisation Service (RISS) is aimed at providing energy oprimised routing in networks with a flat topology. It features very low processing overhead to recover a neighbour's clock and delivers a very high precision of time synchronisation. As a result it can be used very efficiently for duty cycling of the nodes and also for time-tamping of the observed events.

The other group of methods in which I proposed to use location information to enhance routing is aimed at preventing various problems related to the harsh environment of wireless transmission and as a result they contribute to the overall energy optimisation of packets routing in WSNs. The first technique, called Smart Clustering Adapted LEACH (SCALE), adjusts the intra-cluster topology so as to optimise the energy efficiency of routing. In many cluster-based routing algorithms nodes join the closest cluster. This decision may result in the formation of an inefficient path to the base station because the packets may travel away from it. I investigate the conditions for which the intra-cluster communication is energy efficient and I incorporate the results of my study into the SCALE protocol. Simulations show that SCALE performs better in terms of energy consumption and system lifetime than the LEACH protocol.

I have also investigated when multi-hopping is more energy efficient than direct transmission, using a standard wireless model. I also established the optimal location of the relay for the two-hop communication scheme. These inferences were verified experimentally on sensor networking hardware and can be incorporated into a routing protocol if it knows the location of nodes.

As a last optimisation method I propose a technique to delineate the transition region. The communication with nodes located in this area may result in high levels of packet loss. I evaluated my method for a 2.4GHz O-QPSK architecture and tested it in various environments. A node can exclude from its routing table neighbours which it has determined are located in a transition region.

5.2 Future work

This thesis provides novel methods for the energy optimisation of routing protocols using a cross-layer design approach. Nevertheless, a number of interesting research problems require further investigation.

Throughout my work I have used both simulation techniques and experiment with sensor networking hardware to test proposed solutions. The experiments were carried with networks of up to 20 nodes. However it would be interesting to assess these methods on a larger scale WSN. This would require having access to a network composed of many compatible sensor nodes. Unfortunately, the amount of networking equipment available for my use in the laboratory does not allow me to create such an environment. One possibility to overcome this obstacle is to deploy the code on a sensornet testbeds *e.g.* MoteLab [149]. The disadvantage of this solution is that it would only enable verification of the robustness of the developed solution but not its energy efficiency. Another possibility is to constitute a community of researchers willing to participate with their equipment in a test network. Such a network could be used not only for this project but also for testing the performance of other prototype routing protocols.

In my future work I would like to test the novel solutions on various sensornets platforms. First of all, the method which can be used to delineate the transition region is designed for 2.4GHz O-QPSK architecture. The 2.4 GHz band is available for WSNs worldwide whereas the 800 MHz and 900 MHz bands are only allowed for these networks in some countries. It would be interesting to test my approach for the other types of WSN transceivers. Also I performed tests of RISS and CLEAR on a platform which has a very fast waking time and thus is very energy efficient when used for duty cycling. I would like to carry tests of these methods with other platforms and if possible in a heterogenous WSN to test the performance of these techniques when sensor nodes and especially their clocks have different characteristics.

5.3 Concluding remarks

The deployment and development of energy efficient routing strategies, although very difficult for currently available hardware sensor nodes, is essential to guarantee the further expansion of WSNs. In this thesis I have presented specific WSN routing issues and current solutions to address such issues. I have also proposed new and more energy efficient routing methods which employ a cross-layer design approach. The use of these techniques may extend the lifetime of the network and also helps to overcome problems in wireless transmission due to unreliable links.

These solutions, if applied in a future routing protocol for WSNs, will facilitate the development of energy efficient and reliable communication in these networks.

APPENDIX A

Measurements of energy consumption

One of the metrics used during my experiment was the energy efficiency of the protocol. This can be expressed and measured in many ways. Experiments showed that among all a sensor node's components it is the transceiver that drains most of the power from the battery. Hence, there are two ways of comparing the energy efficiency of protocols:

- Compare an average power consumption of a node for considered protocols
- Compare an average awake time of the transceiver for examined protocols

In this appendix I describe how measurements of energy efficiency were performed.

A.1 Measurement of average power consumption by the sensor node

To measure an average power consumption by the sensor node I built the circuit depicted in figure A.1.

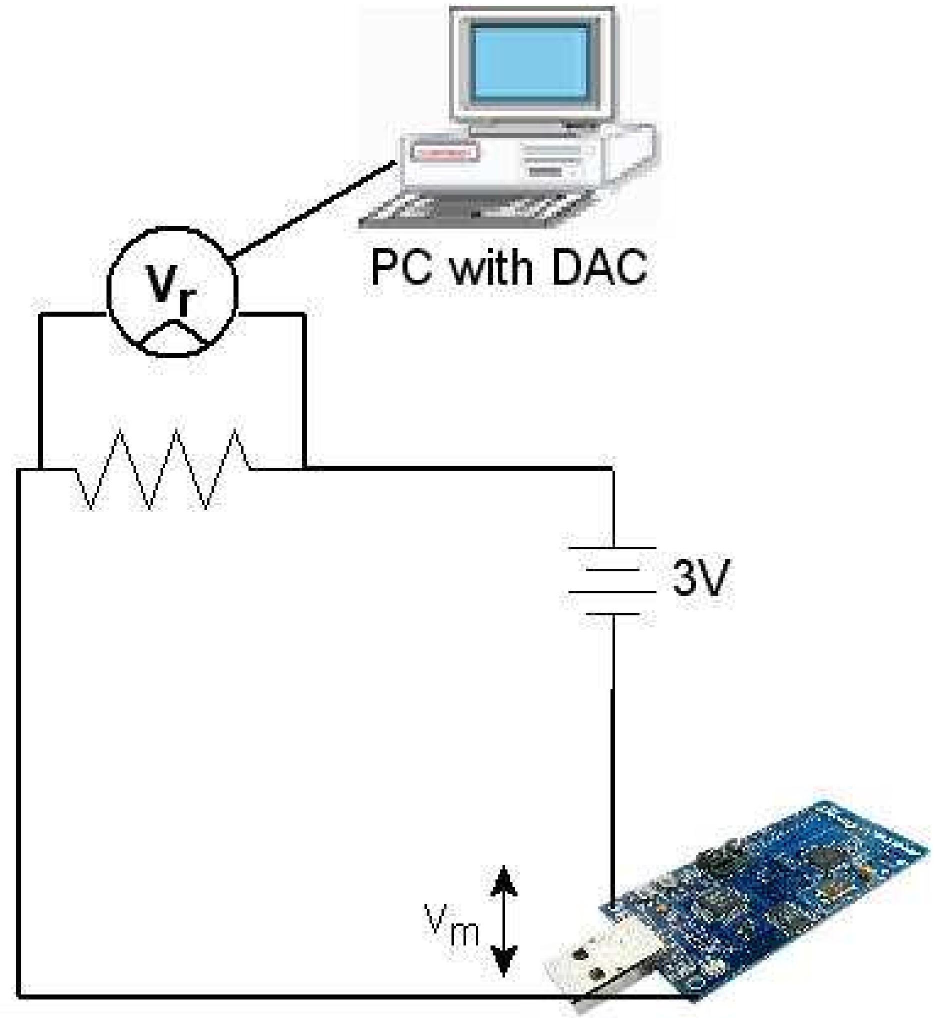


Figure A.1: *Scheme of the power consumption circuit.*

I performed my experiment on the Tmote Sky platform from Moteiv[122]. The Tmote Sky module is a low power "mote" with integrated sensors, radio, antenna, microcontroller, and programming capabilities. To be powered it requires a voltage source between 2.7 and 3.6V. In order to measure its power consumption I connected in series a Tmote Sky with a 1Ω resistance. The circuit is closed with a 3V regulated power supply. The power consumption of the sensor node can be deduced from the measurements of the V_r voltage on the resistance because TmoteSky and the resistence are connected in series. So, the same current is drained by the sensor node and flows through the resistance. The power consumption of the TmoteSky can be expressed as:

$$P = V_m \times I_m \tag{A.1}$$

where V_m is the voltage measured across the sensor node and I_m is the current which flows through it. These values can be deduced from the measurement of voltage V_r across the 1Ω resistance. The current I_m can be obtained with the equation:

$$I_m = \frac{V_r}{R} \tag{A.2}$$

where R equals 1Ω. Finally the power consumption of the TmoteSky is:

$$P = V_m \times \frac{V_r}{R} = (3 - V_r) \times \frac{V_r}{R} = \frac{3V_r}{R} - \frac{V_r^2}{R} \tag{A.3}$$

Since my interest was in comparative rather than absolute measures of power consumption, it was not felt necessary to use a high precision 1Ω resistor and a resistor of standard precision was used.

According to TmoteSky specification [132] maximum value of current I_m can be 20mA. Thus the maximum measured value of V_r can be 20mV which can be deduced from equation A.2 where $R = 1\Omega$. Hence, from equation A.3 I can approximate the value P of power consumed by

$$P \approx \frac{3V_r}{R} \tag{A.4}$$

which results in maximal relative error of 0.67%. The relative error and difference between value of P (calculated with equation A.3) and its approximation with equation A.4 are shown in figure A.3.

Hence, to obtain a power consumption of the sensor node I measured voltage V_r using NI 5112 High-Speed digitiser [109]. The measured voltage was displayed with a LabVIEW application developed for this purpose. With the same program I could store measured samples for further treatment. In this way I could for example calculate an average power consumption. A typical screenshot of the voltage measurements is shown in figure A.2. The apparent negative values are the result of software errors when displaying a large number of collected samples (10MHz sampling rate). The same High-Speed digitiser and the LabVIEW application were used to measure average awake time of the transceiver.

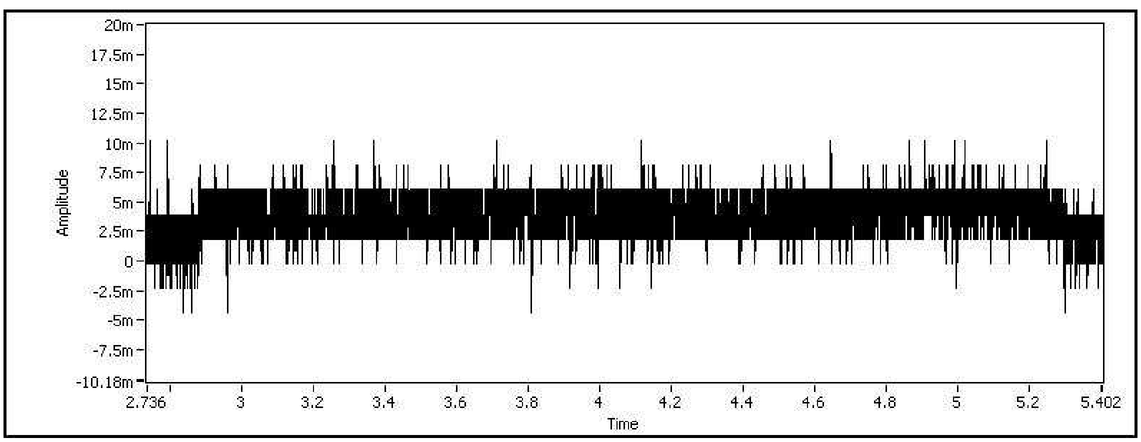
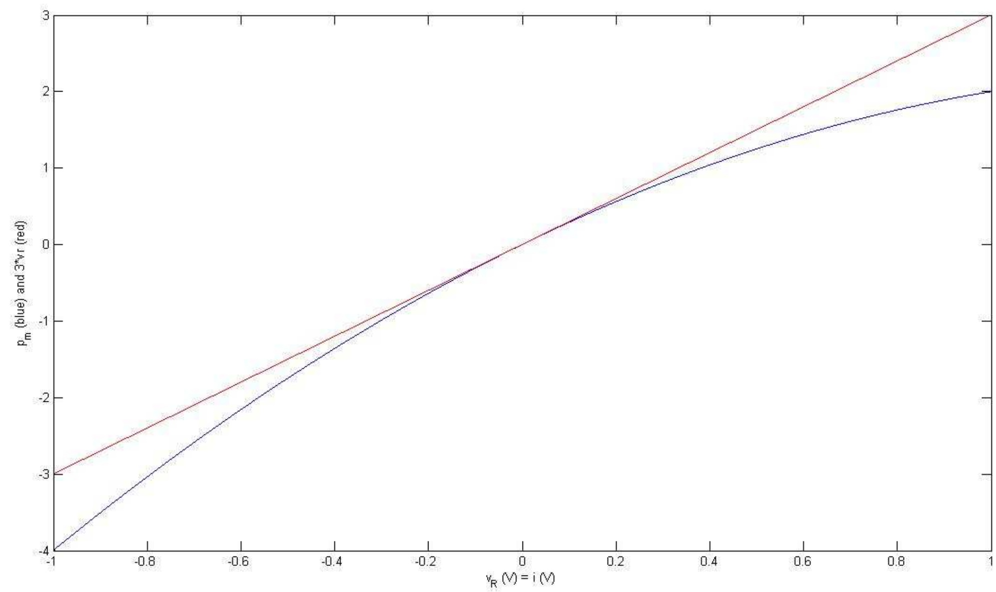


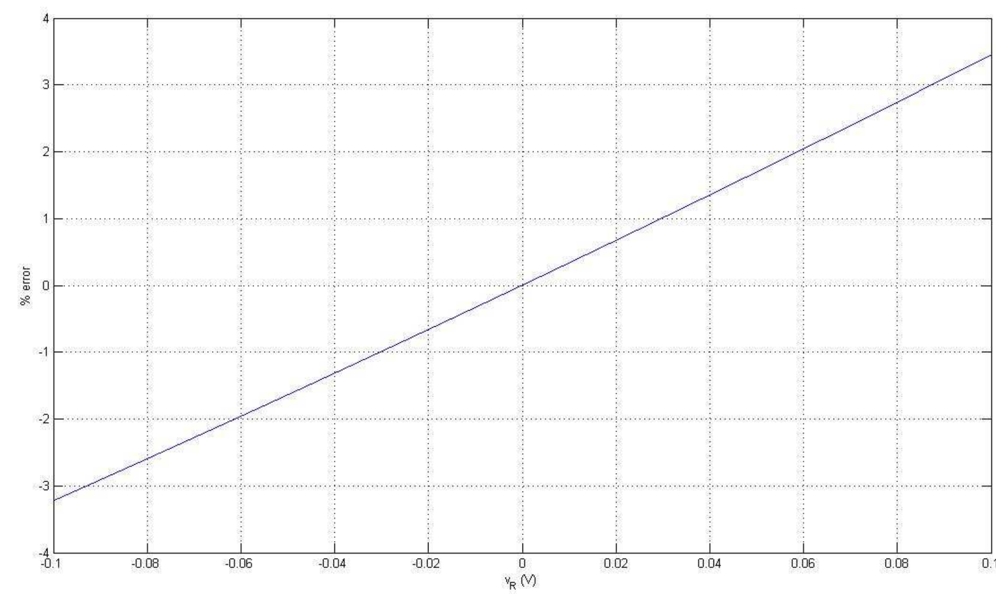
Figure A.2: *Sample of voltage measurement across resistance R.*

A.2 Measurement of average awake time of the transceiver

In my experiments I measured the awake time of the transceiver to compare energy efficiency of different protocols. These measurements were performed in



(a)



(b)

Figure A.3: *Approximation error of power consumption measurement: (a) difference between P_m calculated with equation A.3 (blue) and approximated with equation A.4 (red) in function of V_r (b) relative error of this approximation in function of V_r.*

software but to verify these results I estimated the awake time using the NI 5112 High-Speed digitiser and LabVIEW application previously described. Whenever the transceiver was awake the application increased the voltage of a user general I/O pin available on the TmoteSky. After an experiment I measured the time when the pin was at the higher voltage level and compared that with awake time measurement provided by software estimation.

APPENDIX B

Model of transition region

Zuniga *et al.* claim that the transitional region is the result of placing specific devices in specific environment [163] and thus there are three components which influence the extent of the transition region: the wireless channel, radio transmission characteristics, and the noise floor.

The wireless channel model used by the author is one of the most common radio propagation models, the so-called the log-normal shadowing path loss model, as described by Rappaport [126]:

$$P_r(d) = P_t - PL(d_o) - 10\,\eta\,\log_{10}\left(\frac{d}{d_o}\right) + \mathcal{N}(0, \sigma) \tag{B.1}$$

where $P_r(d)$ is the received power at distance d, P_t is the transmit power, $PL(d_0)$ is the power lost at a reference distance d_0, η is the decay factor, and σ is the standard deviation of the shadowing component.

Zuniga analyses the transition region model for radio using NRZ encoding and a non-coherent FSK modulation scheme. He claims also that it can be extended to other types of radio architecture. The radio model is based on the work of Lal *et al.* [41] where the probability p of successfully receiving a packet is given by:

$$p = (1 - P_e)^{8l}(1 - P_e)^{8(f-l)} = (1 - P_e)^{8f} \tag{B.2}$$

where P_e is the probability of bit error, f the frame size, and l the preamble length (both expressed in bytes). Furthermore, p can be expressed as:

$$p = (1 - \frac{1}{2}\exp^{-\frac{\alpha}{2}})^{8f} \tag{B.3}$$

because for non-coherent FSK modulation, P_e is given by:

$$P_e = \frac{1}{2}\exp^{-\frac{\alpha}{2}} \tag{B.4}$$

with $\alpha = E_b/N_o$. Since most commercial radios do not provide the $\alpha = E_b/N_o$ metric, Zuniga proposes to integrate the SNR (Signal-to-Noise Ratio) value into equation B.3. Then the RSSI (Received Signal Strength Indication) measurements can be used to determine SNR. Hence SNR is given by:

$$SNR = \frac{E_b}{N_o}\frac{R}{B_N} \tag{B.5}$$

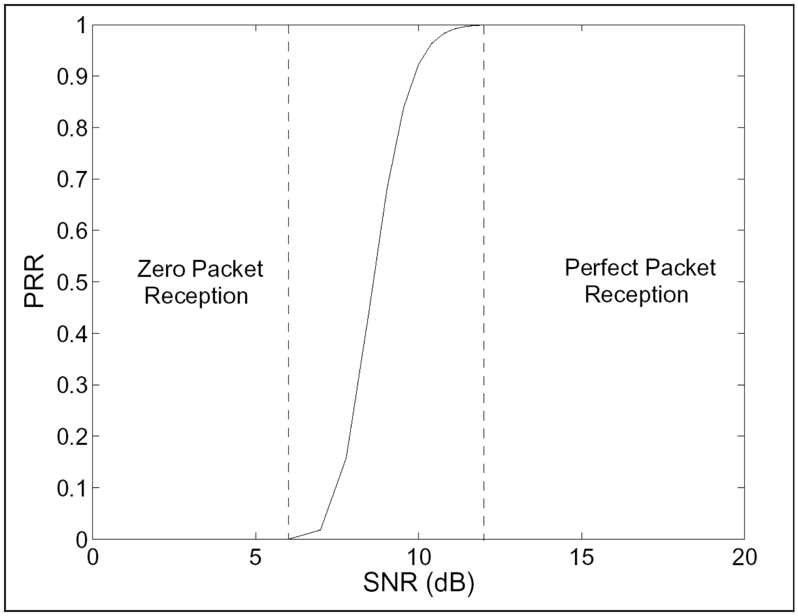
and thus equation B.3 becomes:

$$p = (1 - \frac{1}{2}\exp^{-\frac{\gamma}{2}\frac{B_N}{R}})^{8f} \tag{B.6}$$

where γ is SNR, B_N the noise bandwidth, and R the data rate. An example of the radio model plot is shown in figure B.1.

The third element determining the transition region is the noise floor, which depends on both the radio architecture and the environment. For the case when the receiver and antenna have the same ambient temperature the noise floor can be determined analytically using the equation [126]:

$$P_n = FkT_oB \tag{B.7}$$



Source: Analyzing the transitional region in low power wireless links [163]

Figure B.1: *Radio model: Non-coherent FSK, NRZ radio, f=50 bytes, data rate=19.2kbps, noise bandwidth=30kHz.*

where F is the noise figure of the radio, k the Boltzmann's constant, T_o is the ambient temperature, and B is the noise equivalent bandwidth.

Integrating all these three elements can lead to an expression of transition region parameters as follows. The SNR γ at distance d can be estimated with the equation:

$$\gamma(d)_{dB} = P_{t\ dB} - PL(d)_{dB} - P_{n\ dB} \tag{B.8}$$

where P_t is the transmitting power. This equation can be incorporated into equation B.6 and thus the PRR at a distance d is given by:

$$p(d) = (1 - \frac{1}{2}\exp^{-\frac{\gamma(d)}{2}\frac{B_N}{R}})^{8f} \tag{B.9}$$

With the aim of expressing the radius of different regions, Zuniga proposes to bound the connected region to PRR greater than 0.9, and the transition region to values between 0.9 and 0.1. Hence, the corresponding values of SNR, γ_U and γ_L, for PRRs of 0.9 and 0.1 respectively, can be obtained from equation B.9 as follows:

$$\gamma_U \; dB = 10 \log_{10}(-2\frac{R}{B_N} \ln(2(1 - 0.9^{\frac{1}{8f}}))) \tag{B.10}$$

$$\gamma_L \; dB = 10 \log_{10}(-2\frac{R}{B_N} \ln(2(1 - 0.1^{\frac{1}{8f}}))) \tag{B.11}$$

These bounds can be incorporated into the channel model to express the radius of different regions at the link layer. The authors claim that due to the gaussian characteristic of log-normal shadowing in the path loss model, the received signal strength P_r can be bounded within $\pm 2\sigma$, *i.e.* $P(\mu - 2\sigma < P_r < \mu + 2\sigma) = 0.955$. Hence, the received power P_r at a distance d is bonded by:

$$P_{rU}(d) = P_t - \overline{PL}(d) + 2\sigma \tag{B.12}$$

$$P_{rL}(d) = P_t - \overline{PL}(d) - 2\sigma \tag{B.13}$$

where $\overline{PL}(d) = PL(d_o) + 10\,\eta\,\log_{10}\left(\frac{d}{d_o}\right)$.

The transition region begins when the P_r values (P_{rL}) enter the $P_n + \gamma_U$ limit, and ends when the P_r values (P_{rU}) leave the $P_n + \gamma_L$ limit, as shown in figure B.2. Hence we can write that:

$$P_{rU} = \gamma_L + P_n \tag{B.14}$$

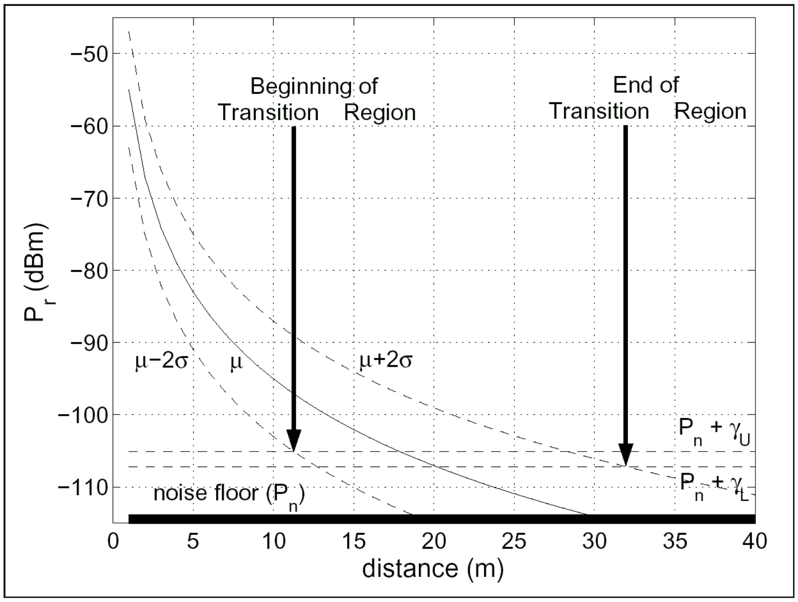
$$P_{rL} = \gamma_U + P_n \tag{B.15}$$

Finally, by integrating equations B.14, B.15 and equations B.12, B.13, we can obtain the following expressions to calculate the extent of the transition region:

$$d_s = 10^{\frac{P_n - \gamma_U - P_t + PL(d_0) + 2\sigma}{-10\eta}} \tag{B.16}$$

$$d_e = 10^{\frac{P_n - \gamma_L - P_t + PL(d_0) - 2\sigma}{-10\eta}} \tag{B.17}$$

where d_s and d_e are the start and end points of the transition region. In order



Source: Analyzing the transitional region in low power wireless links [163]

Figure B.2: *Transition region determined analytically.*

to compare different experimental setups Zuniga also defines another expression which is the ratio of the radius of the transitional and connected regions:

$$\Gamma = \frac{d_e - d_s}{d_s} = 10^{\frac{(\gamma_U - \gamma_L)+4\sigma}{10\eta}} - 1 \quad \text{(B.18)}$$

This analytical study of the transitional region was verified empirically by Zuniga *et al.* [163] where measurements of PRR are compared with the theoretical predictions of transition region bounds. However, this model was only confirmed using a radio architecture that employs NRZ and non-coherent FSK and it may be difficult to directly apply with systems employing more complex modulation schemes, such as O-QPSK. This is because the probability of bit error can be easily expressed in function of E_b/N_o for non-coherent FSK modulation (equation B.4). For other modulation schemes (for example O-QPSK) this figure may be more difficult to obtain.

BIBLIOGRAPHY

[1] *The 29 Palms Experiment: Tracking Vehicles with a UAV-Delivered Sensor Network.* http://robotics.eecs.berkeley.edu/ pister/29Palms0103.

[2] *Intermediate system to intermediate system intra-domain routeing exchange protocol.*, tech. report, Digital Equipment Corporation, October 1989.

[3] *2006: Wireless medium access control (mac) and physical layer (phy) specifications for low rate wireless personal area networks (lr-wpans)*, IEEE std. 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003), (2006).

[4] *VDC's 2007 embedded systems market statistics*. Venture Development Corporation, Dec. 2007.

[5] R. Aaron and R. Wyndrum, *Future trends*, IEEE Communications Magazine, 24 (1986), pp. 38– 43.

[6] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, New York, 1964.

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

[8] J. Al-Karaki and A. Kamal, *Routing techniques in wireless sensor networks: a survey*, Wireless Communications, IEEE, 11 (Dec. 2004), pp. 6–28.

[9] P. Alfke, *Efficient shift registers, lfsr counters, and long pseudo-random sequence generators.*, tech. report, Xilinx, July 1996.

[10] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, M. Stemm, E. Amir, and R. H. Katz, *TCP Improvements for Heterogeneous Networks:*

The Daedalus Approach, in Proceedings of 35th Annual Allerton Conference on Communication, Control, Urbana, IL, Oct. 1997.

[11] M. BARR, *Programming Embedded Systems in C and C++*, O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1998.

[12] A. BARROSO, U. ROEDIG, AND C. SREENAN, *μ-mac: an energy-efficient medium access control for wireless sensor networks*, Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on, (31 Jan.-2 Feb. 2005), pp. 70–80.

[13] A. BARROSO, U. ROEDIG, AND C. J. SREENAN, *Use of framelets for efficient transmitter-receiviver rendezvous in wireless sensor networks*, The IEEE Conference on Local Computer Networks., (15-17 Nov. 2005), pp. 664–671.

[14] R. E. BELLMAN, *Dynamic Programming*, Princeton University Press, 1957.

[15] R. BERNHARDT, *Macroscopic diversity in frequency reuse radio systems*, Selected Areas in Communications, IEEE Journal on, 5 (Jun 1987), pp. 862–870.

[16] BOOMERANG, *Moteiv corporation. www.moteiv.com/software*.

[17] G. BORRIELLO AND R. WANT, *Embedded computation meets the world wide web*, Commun. ACM, 43 (2000), pp. 59–66.

[18] D. BRAGINSKY AND D. ESTRIN, *Rumor routing algorthim for sensor networks*, in WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, New York, NY, USA, 2002, ACM, pp. 22–31.

[19] A. CARZANIGA AND A. L. WOLF, *Content-based networking: A new communication infrastructure*, in IMWS '01: Revised Papers from the NSF Workshop on Developing an Infrastructure for Mobile and Wireless Systems, London, UK, 2002, Springer-Verlag, pp. 59–68.

[20] B. CHEN, K. JAMIESON, H. BALAKRISHNAN, AND R. MORRIS, *Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks*, in MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking, New York, NY, USA, 2001, ACM, pp. 85–96.

[21] D. Chen and P. K. Varshney, *Qos support in wireless sensor networks: A survey*, in ICWN 2004: Proceedings of the 2004 International Conference on Wireless Networks (ICWN 2004), Las Vegas, Nevada, USA, 2004, p. 116.

[22] Y. Chen, E. G. Sirer, and S. B. Wicker, *On selection of optimal transmission power for ad hoc networks*, System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on, (6-9 Jan. 2003), p. 10.

[23] Chipcon, *2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*, texas instruments literature no: swrs041b. ed.

[24] ——, *CC2431 Location Engine, Application Note AN042*, Texas Instruments Literature No: swra095. ed.

[25] M. Chu, H. Haussecker, and F. Zhao, *Scalable information-driven sensor querying and routing for Ad Hoc Heterogeneous Sensor Networks*, j-IJHPCA, 16 (2002), pp. 293–313.

[26] CIRRONET, *RF Power Options in ZigBee Solutions*, cirronet inc. literature ed., 2005.

[27] D. D. Clark, *Modularity and efficiency in protocol implementation*, 1982.

[28] D. D. Clark and D. L. Tennenhouse, *Architectural considerations for a new generation of protocols*, in SIGCOMM '90: Proceedings of the ACM symposium on Communications architectures & protocols, New York, NY, USA, 1990, ACM, pp. 200–208.

[29] T. Clausen and P. Jacquet, *Optimized Link State Routing Protocol (OLSR)*. Published Online, http://rfc.net/rfc3626.txt, October 2003.

[30] M. K. Cox, D. C. and A. Norris, *800 mhz attenuation measured in and around suburban houses*, AT&T Bell Laboratory Technical Journal, 673 (1984).

[31] E. S. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, *RFC 2386: A framework for qos-based routing in the internet*, Aug 1998.

[32] R. L. Cruz and A. Santhanam, *Optimal Routing, Link Scheduling , and Power Control in Multi-hop Wireless Networks*, in INFOCOM, 2003.

[33] S. Cui, R. Madan, A. J. Goldsmith, and S. Lall, *Cross-Layer Energy and Delay Optimization in Small-Scale Sensor Networks*, Wireless Communications, IEEE Transactions on, 6 (October 2007), pp. 3688–3699.

[34] D. CULLER, P. DUTTA, C. T. EE, R. FONSECA, J. HUI, P. LEVIS, J. POLASTRE, S. SHENKER, I. STOICA, G. TOLLE, AND J. ZHAO, *Towards a sensor network architecture: lowering the waistline*, in HOTOS'05: Proceedings of the 10th conference on Hot Topics in Operating Systems, Berkeley, CA, USA, 2005, USENIX Association, pp. 24–24.

[35] D. CULLER, D. ESTRIN, AND M. SRIVASTAVA, *Guest editors' introduction: Overview of sensor networks*, Computer, 37 (2004), pp. 41–49.

[36] D. CULLER AND H. MULDER, *Smart sensors to network the world*, Scientific American, 6 (June 2004), pp. 84–91.

[37] S. B. G. J. D. AGUAYO, J. BICKET AND R. MORRIS, *Link-level measurements from an 802.11b mesh network*, SIGCOMM Comput. Commun. Rev., 34 (2004), pp. 121–132.

[38] B. K. D. SON AND J. HEIDEMANN, *Experimental analysis of concurrent packet transmissions in low-power wireless networks.*, tech. report, ISI-TR-2005-609, Nov. 2005.

[39] A. F. DANA AND B. HASSIBI, *On the power efficiency of sensory and ad hoc wireless networks*, IEEE Transactions on Information Theory, 52 (2006), pp. 2890–2914.

[40] J. DENG, Y. S. HAN, P.-N. CHEN, AND P. K. VARSHNEY, *Optimum transmission range for wireless ad hoc networks*, Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE, 2 (21-25 March 2004), pp. 1024–1029 Vol.2.

[41] L. DHANANJAY, A. MANJESHWAR, F. HERRMANN, E. UYSAL-BIYIKOGLU, AND A. KESHAVARZIAN, *Measurement and characterization of link quality metrics in energy constrained wireless sensor networks*, Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE, 1 (1-5 Dec. 2003), pp. 446–452 Vol.1.

[42] R. DRAVES, J. PADHYE, AND B. ZILL, *Routing in multi-radio, multi-hop wireless mesh networks*, in MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking, New York, NY, USA, 2004, ACM, pp. 114–128.

[43] A. EPHREMIDES, *Energy concerns in wireless networks*, Wireless Communications, IEEE, 9 (Aug. 2002), pp. 48–59.

[44] D. ESTRIN, R. GOVINDAN, J. HEIDEMANN, AND S. KUMAR, *Next century challenges: scalable coordination in sensor networks*, in MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, New York, NY, USA, 1999, ACM, pp. 263–270.

[45] M. ETTUS, *System capacity, latency, and power consumption in multihop-routed SS-CDMA wireless networks*, Radio and Wireless Conference, 1998. RAWCON 98. 1998 IEEE, (9-12 Aug 1998), pp. 55–58.

[46] S. FEDOR AND M. COLLIER, *On the problem of energy efficiency of multi-hop vs one-hop routing in wireless sensor networks*, in AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops, Washington, DC, USA, 2007, IEEE Computer Society, pp. 380–385.

[47] W. FEIBEL, *The encyclopedia of networking*, San Francisco : Alameda, CA : Network Press, 2nd ed., 1996.

[48] A. FESTAG, *Optimization of Handover Performance by Link Layer Triggers in IP-Based Networks; Parameters, Protocol Extensions, and APIs for Implementation*, Tech. Report TKN-02-014, Telecommunication Networks Group, Technische Universität Berlin, July 2002.

[49] D. GANESAN, B. KRISHNAMACHARI, A. WOO, D. CULLER, D. ESTRIN, AND S. WICKER, *Complex behavior at scale: An experimental study of low-power wireless sensor networks*, Tech. Report CSD-TR 02-0013, UCLA, February 2002.

[50] A. GOLDSMITH AND A. GOLDSMITH, *Wireless Communications*, Cambridge University Press, New York, NY, USA, 2005.

[51] Z. HAAS, *A new routing protocol for the reconfigurable wireless networks*, 1997.

[52] M. HAENGGI, *Routing in ad hoc networks: A case for long hops*, IEEE Communications Magazine, (2004). Submitted to Series on Ad Hoc and Sensor Networks.

[53] ——, *Routing in ad hoc networks-a wireless perspective*, in BROADNETS 04: Proceedings of the First International Conference on Broadband Networks, Washington, DC, USA, 2004, IEEE Computer Society, pp. 652–660.

[54] ——, *Twelve reasons not to route over many short hops*, in IEEE Vehicular Technology Conference (VTC04 Fall), Los Angeles, CA, SEP 2004.

[55] V. HANDZISKI, H. KARL, A. KÖPKE, AND A. WOLISZ, *A common wireless sensor network architecture?*, in Proc. 1. GI/ITG Fachgespräch "Sensornetze" (Technical Report TKN-03-012 of the Telecommunications Networks Group, Technische Universität Berlin, H. Karl, ed., Berlin, Germany, Jul 2003, pp. 10–17.

[56] M. HATA, *Empirical formula for propagation loss in land mobile radio services*, Vehicular Technology, IEEE Transactions on, 29 (Aug 1980), pp. 317–325.

[57] T. HE, J. A. STANKOVIC, C. LU, AND T. ABDELZAHER, *SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks*, in ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems, Washington, DC, USA, 2003, IEEE Computer Society, p. 46.

[58] S. M. HEDETNIEMI, S. T. HEDETNIEMI, AND A. L. LIESTMAN, *A Survey of Gossiping and Broadcasting in Communication Networks*, NETWORKS: Networks: An International Journal, 18 (1988).

[59] C. L. HEDRICK, *Routing information protocol*, 1988.

[60] J. HEIDEMANN, N. BULUSU, J. ELSON, C. INTANAGONWIWAT, K. CHAN LAN, Y. XU, W. YE, D. ESTRIN, AND R. GOVINDAN, *Effects of detail in wireless network simulation*, in SCS Communication Networks and Distributed Systems Modeling and Simulation Conference, Phoenix, AZ, January 2001, pp. 1–10.

[61] W. R. HEINZELMAN, A. P. CHANDRAKASAN, AND H. BALAKRISHNAN, *Energy-Efficient Communication Protocol for Wireless Microsensor Networks*, in HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8, Washington, DC, USA, 2000, IEEE Computer Society, p. 8020.

[62] ——, *An application-specific protocol architecture for wireless microsensor networks*, Wireless Communications, IEEE Transactions on, 1 (Oct 2002), pp. 660–670.

[63] W. R. HEINZELMAN, J. KULIK, AND H. BALAKRISHNAN, *Adaptive protocols for information dissemination in wireless sensor networks*, in MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, New York, NY, USA, 1999, ACM, pp. 174–185.

[64] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, *System architecture directions for networked sensors*, in ASPLOS-IX: Proceedings of the ninth international conference on Architectural support for programming languages and operating systems, New York, NY, USA, 2000, ACM, pp. 93–104.

[65] J. L. Hill and D. E. Culler, *Mica: A Wireless Platform for Deeply Embedded Networks*, IEEE Micro, 22 (2002), pp. 12–24.

[66] T.-C. Hou and V. Li, *Transmission range control in multihop packet radio networks*, Communications, IEEE Transactions on, 34 (Jan 1986), pp. 38–44.

[67] I. Howitt and J. Wang, *Energy balanced chain in distributed sensor networks*, Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE, 3 (21-25 March 2004), pp. 1721–1726 Vol.3.

[68] B. Hurley, C. Seidl, and W. Sewell, *A survey of dynamic routing methods for circuit-switched traffic*, IEEE Communications Magazine, 25 (1987), pp. 13– 21.

[69] M. Ilyas and I. Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*, CRC, July 2004.

[70] C. Intanagonwiwat, R. Govindan, and D. 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, New York, NY, USA, 2000, ACM, pp. 56–67.

[71] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, *Directed diffusion for wireless sensor networking*, IEEE/ACM Trans. Netw., 11 (2003), pp. 2–16.

[72] Q. Jiang and D. Manivannan, *Routing protocols for sensor networks*, in CCNC '04: Proceedings of the IEEE Consumer Communications and Networking Conference, Las Vegas, Nevada, USA, 2004.

[73] X. Jiu-qiang, S. Pei-gang, Z. Hai, Z. Xi-yuan, Z. Jian, and C. Dawei, *An analytical model for link evaluation of wireless sensor networks*, Network and Parallel Computing Workshops, 2007. NPC Workshops. IFIP International Conference on, (18-21 Sept. 2007), pp. 386–391.

[74] M. Joa-Ng and I.-T. Lu, *A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks*, Selected Areas in Communications, IEEE Journal on, 17 (Aug 1999), pp. 1415–1425.

[75] A. E. JOEL, ed., *Asynchronous Transfer Mode Switching*, Institute of Electrical & Electronic Engineering, 1993.

[76] D. B. JOHNSON AND D. A. MALTZ, *Dynamic Source Routing in Ad Hoc Wireless Networks*, in Mobile Computing, Imielinski and Korth, eds., vol. 353, Kluwer Academic Publishers, 1996.

[77] J. JUBIN AND J. D. TORNOW, *The DARPA packet radio network protocols.*, Proceedings of the IEEE, (1987), pp. 21–32.

[78] R. JURDAK., *Modeling and Optimization of Ad Hoc and Sensor Networks*, PhD thesis, Bren School of Information and Computer Science, University of California Irvine, September 2005.

[79] H. KARL AND A. WILLIG, *Protocols and Architectures for Wireless Sensor Networks*, John Wiley & Sons, 2005.

[80] C. KARLOF AND D. WAGNER, *Secure routing in wireless sensor networks: attacks and countermeasures*, Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on, (11 May 2003), pp. 113–127.

[81] B. KARP AND H. T. KUNG, *GPSR: greedy perimeter stateless routing for wireless networks*, in MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking, New York, NY, USA, 2000, ACM, pp. 243–254.

[82] V. KAWADIA AND P. KUMAR, *A cautionary perspective on cross-layer design*, Wireless Communications, IEEE, 12 (Feb. 2005), pp. 3–11.

[83] P. B. KEY AND G. A. COPE, *Distributed dynamic routing schemes*, IEEE Communications Magazine, 28 (1990).

[84] A. KÖPKE, V. HANDZISKI, J.-H. HAUER, AND H. KARL, *Structuring the information flow in component-based protocol implementations for wireless sensor nodes*, in Proc. of Work-in-Progress Session of the 1st European Workshop on Wireless Sensor Networks (EWSN), Technical Report TKN-04-001 of Technical University Berlin, Telecommunication Networks Group, Berlin, Germany, Jan 2004, pp. 41–45.

[85] D. KOTZ, C. NEWPORT, AND C. ELLIOTT, *The mistaken axioms of wireless-network research*, tech. report, Dartmouth College, July 2003.

[86] J. KULIK, W. R. HEINZELMAN, AND H. BALAKRISHNAN, *Negotiation-based protocols for disseminating information in wireless sensor networks*, Wirel. Netw., 8 (2002), pp. 169–185.

[87] B. KUSY, P. DUTTA, P. LEVIS, M. MARÓTI, ÁKOS LÉDECZI, AND D. CULLER, *Elapsed time on arrival: A simple, versatile, and scalable primitive for canonical time synchronization services*, International Journal of Ad Hoc and Ubiquitous Computing, 1 (2006), pp. 239–251.

[88] P. LEVIS, *TinyOS Programming*, June 2006.

[89] N.-C. LIANG, P.-C. CHEN, T. SUN, G. YANG, L.-J. CHEN, AND M. GERLA, *Impact of node heterogeneity in zigbee mesh network routing*, Systems, Man and Cybernetics, 2006. SMC '06. IEEE International Conference on, 1 (8-11 Oct. 2006), pp. 187–191.

[90] S. LINDSAY, C. S. RAGHAVENDRA, AND K. M. SIVALINGAM, *Data gathering in sensor networks using the energy delay metric*, in IPDPS '01: Proceedings of the 15th International Parallel & Distributed Processing Symposium, Washington, DC, USA, 2001, IEEE Computer Society, p. 188.

[91] S. LINDSEY AND C. S. RAGHAVENDRA, *PEGASIS: Power-Efficient Gathering in Sensor Information Systems*, 2002.

[92] B. LISKOV, *Practical uses of synchronized clocks in distributed systems*, in PODC '91: Proceedings of the tenth annual ACM symposium on Principles of distributed computing, New York, NY, USA, 1991, ACM, pp. 1–9.

[93] R. MADAN, S. CUI, S. LALL, AND A. J. GOLDSMITH, *Cross-layer design for lifetime maximization in interference-limited wireless sensor networks*, in INFOCOM, 2005, pp. 1964–1975.

[94] A. MANJESHWAR AND D. P. AGRAWAL, *TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks*, in IPDPS '01: Proceedings of the 15th International Parallel & Distributed Processing Symposium, Washington, DC, USA, 2001, IEEE Computer Society, p. 189.

[95] ——, *APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks*, in IPDPS '02: Proceedings of the 16th International Parallel and Distributed Processing Symposium, Washington, DC, USA, 2002, IEEE Computer Society, p. 48.

[96] N. Z. MARCO ZÚ AND B. KRISHNAMACHARI, *An analysis of unreliability and asymmetry in low-power wireless links*, ACM Trans. Sen. Netw., 3 (2007), p. 7.

[97] M. MARÓTI, B. KUSY, G. SIMON, AND ÁKOS LÉDECZI, *The flooding time synchronization protocol*, in SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems, New York, NY, USA, 2004, ACM, pp. 39–49.

[98] P. J. MARRÓN, D. MINDER, A. LACHENMANN, AND K. ROTHERMEL, *TinyCubus: An adaptive cross-layer framework for sensor networks*, IT - Information Technology, 47 (2005), pp. 87–97.

[99] J. MARTIN, K. K. CHAPMAN, AND C. I. T. A. GROUP, *SNA: IBM's networking solution*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1987.

[100] J. M. MCQUILLAN, I. RICHER, AND E. C. ROSEN, *The new routing algorithm for the arpanet*, IEEE Transactions on Communications, 28 (1988), pp. 711–719.

[101] J. M. MCQUILLAN AND D. C. WALDEN, *ARPA network design decisions*, j-COMP-NET, 1 (1977), pp. 243–289.

[102] T. H. MENG AND V. RODOPLU, *Distributed network protocols for wireless communication*, Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on, 4 (31 May-3 Jun 1998), pp. 600–603 vol.4.

[103] V. MHATRE AND C. ROSENBERG, *Homogeneous vs heterogeneous clustered sensor networks: a comparative study*, Communications, 2004 IEEE International Conference on, 6 (20-24 June 2004), pp. 3646–3651 Vol.6.

[104] R. MIN AND A. CHANDRAKASAN, *Mobicom poster: top five myths about the energy consumption of wireless communication*, SIGMOBILE Mob. Comput. Commun. Rev., 7 (2003), pp. 65–67.

[105] P. MOHAPATRA AND S. KRISHNAMURTHY, *Ad Hoc Networks: Technologies and Protocols*, Springer, New York, NY, USA, 2005.

[106] G. MOORE, *Cramming more components onto integrated circuits*, Proceedings of the IEEE, 86 (Jan 1998), pp. 82–85.

[107] J. MOY, *Ospf version 2*. RFC 2328 (Standard), apr 1998.

[108] S. D. MURUGANATHAN, D. C. F. MA, R. I. BHASIN, AND A. O. FAPOJUWO, *A centralized energy-efficient routing protocol for wireless sensor networks*, Communications Magazine, IEEE, 43 (March 2005), pp. S8–13.

[109] National Instruments, *NI 5112 User Manual*, national instruments literature part number: 322628b-01 . ed., November 2000.

[110] J. V. Neumann, A. W. Taub, and A. H. Taub, *The Collected Works of John von Neumann: 6-Volume Set*, Reader's Digest Young Families, 1963.

[111] N. Nikaein, C. Bonnet, and N. Nikaein, *HARP - Hybrid ad hoc routing protocol*, in IST 2001, International symposium on telecommunications, September 1-3 2001, Teheran, Iran, Sep 2001.

[112] N. Ouferhat and A. Mellouk, *Optimal qos and adaptatiw routing in wireless sensor networks*, Information and Communication Technologies, 2006. ICTTA '06. 2nd, 2 (24-28 April 2006), pp. 2736–2741.

[113] M. A. Padlipsky, *RFC 871: Perspective on the ARPANET reference model*, sep 1982. Status: UNKNOWN.

[114] V. D. Park and M. S. Corson, *A highly adaptive distributed routing algorithm for mobile wireless networks*, in INFOCOM '97: Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution, Washington, DC, USA, 1997, IEEE Computer Society, p. 1405.

[115] G. Pei, M. Gerla, and T.-W. Chen, *Fisheye State Routing in Mobile Ad Hoc Networks*, in ICDCS Workshop on Wireless Networks and Mobile Computing, 2000, pp. D71–D78.

[116] M. Perillo, Z. Cheng, and W. R. Heinzelman, *On the problem of unbalanced load distribution in wireless sensor networks*, Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004. IEEE, (29 Nov.-3 Dec. 2004), pp. 74–79.

[117] C. Perkins, *Ad-hoc on-demand distance vector routing*, 1997.

[118] C. Perkins and P. Bhagwat, *Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers*, in ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications, 1994, pp. 234–244.

[119] D. G. Pery, H. Blumenthal, and R. M. Hinden, *The ARPANET and the DARPA internet*, Libr. Hi Tech, 6 (1988), pp. 51–62.

[120] J. Polastre, J. Hill, and D. Culler, *Versatile low power media access for wireless sensor networks*, in SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems, New York, NY, USA, 2004, ACM, pp. 95–107.

[121] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, and I. Stoica, *A unifying link abstraction for wireless sensor networks*, in SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems, New York, NY, USA, 2005, ACM Press, pp. 76–89.

[122] J. Polastre, R. Szewczyk, and D. E. Culler, *Telos: enabling ultra-low power wireless research.*, in IPSN, 2005, pp. 364–369.

[123] G. J. Pottie and W. J. Kaiser, *Wireless integrated network sensors*, Commun. ACM, 43 (2000), pp. 51–58.

[124] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical recipes in C: the art of scientific computing*, Cambridge University Press, New York, NY, USA, 1988.

[125] J. Proakis, *Digital Communications*, McGraw-Hill Science/Engineering/-Math, August 2000.

[126] T. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[127] L. Roberts, *The arpanet and computer networks*, in Proceedings of the ACM Conference on The history of personal workstations, New York, NY, USA, 1986, ACM, pp. 51–58.

[128] V. Rodoplu and T. Meng, *Minimum energy mobile wireless networks*, IEEE J. Select. Areas Commun., 17 (1999), pp. 1333 – 1344.

[129] E. C. Rosen, *RFC 827: Exterior Gateway Protocol (EGP)*, oct 1982.

[130] P. M. Ruiz and I. Stojmenovic, *Handbook of Approximation Algorithms and Metaheuristics*, ch. Cost-efficient Multicast Routing in Ad hoc and Sensor Networks.

[131] M. K. Schwartz, *Computer Communications Network Design and Analysis*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1977.

[132] Sentilla Corporation, *Tmote Sky Datasheet.*, 1.04 ed., Nov. 2006. Available from http://www.sentilla.com/moteiv-endoflife.html [Accessed 18 January 2008].

[133] S. D. Servetto and G. Barrenechea, *Constrained random walks on random graphs: routing algorithms for large scale wireless sensor networks*, in WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, New York, NY, USA, 2002, ACM, pp. 12–21.

[134] T. J. Shepard, *A channel access scheme for large dense packet radio networks*, SIGCOMM Comput. Commun. Rev., 26 (1996), pp. 219–230.

[135] J.-P. Sheu, C.-S. Hsu, and Y.-J. Chang, *Efficient broadcasting protocols for regular wireless sensor networks: Research articles*, Wirel. Commun. Mob. Comput., 6 (2006), pp. 35–48.

[136] M. L. Sichitiu, *Cross-layer scheduling for power efficiency in wireless sensor networks*, in INFOCOM, 2004.

[137] M. Sikora, J. Laneman, M. Haenggi, J. Costello, D.J., and T. Fuja, *On the optimum number of hops in linear wireless networks*, Information Theory Workshop, 2004. IEEE, (24-29 Oct. 2004), pp. 165–169.

[138] B. Sklar, *Digital Communications - Fundamentals and Applications*, Prentice-Hall, Upper Saddle River, 2.ed ed., 2001.

[139] G. Smaragdakis, I. Matta, and A. Bestavros, *SEP: A Stable Election Protocol for clustered heterogeneous wireless sensor networks*, tech. report, CS Department, Boston University, May 31 2004.

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

[141] S. Soro and W. Heinzelman, *Prolonging the lifetime of wireless sensor networks via unequal clustering*, Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International, (4-8 April 2005).

[142] C. Sreenan, S. Nawaz, T. Le, and S. Jha, *On the sensitivity of sensor network simulations*, Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd, 3 (7-10 May 2006), pp. 1043–1047.

[143] K. Srinivasan and P. Levis, *RSSI is under appreciated*, in In Proc. of the Third Workshop on Embedded Networked Sensors, EmNets,, May 2006.

[144] M. Steenstrup, ed., *Routing in communications networks*, Prentice Hall International (UK) Ltd., Hertfordshire, UK, 1995.

[145] A. S. Tanenbaum, *Computer networks*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 4th ed. ed., 2003.

[146] G. Thonet and M. Bruel, *Zigbee: The journey toward deployment in industrial applications*, ST Journal of Research, 4 (2007), pp. 65–79.

[147] C. L. Wang, B. Yao, Y. Yang, and Z. Zhu, *A survey of embedded operating system*. University of California, San Diego, 2001.

[148] Y. Wang, X. Liu, and J. Yin, *Requirements of quality of service in wireless sensor network*, in ICNICONSMCL '06: Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06), Washington, DC, USA, 2006, IEEE Computer Society, p. 116.

[149] P. W. M. Werner-Allen, G.; Swieskowski, *Motelab: a wireless sensor network testbed*, Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on, (15 April 2005), pp. 483–488.

[150] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, *On the construction of energy-efficient broadcast and multicast trees in wireless networks*, INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2 (2000), pp. 585–594 vol.2.

[151] Y. Wu, P. A. Chou, Q. Zhang, K. Jain, W. Zhu, and S.-Y. Kung, *Network planning in wireless ad hoc networks: a cross-layer approach*, Selected Areas in Communications, IEEE Journal on, 23 (Jan. 2005), pp. 136–150.

[152] Y. Xu, J. Heidemann, and D. Estrin, *Geography-informed energy conservation for Ad Hoc routing*, in MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking, New York, NY, USA, 2001, ACM, pp. 70–84.

[153] Y. Okumura, E. Ohmori, T. Kawano, and K. Fukuda, *Field Strength and its Variability in VHF and UHF Land Mobile Service*, Rev. Elec. Comm. Lab, 16 (1968), p. 825.

[154] F. Ye, A. Chen, S. Lu, and L. Zhang, *A scalable solution to minimum cost forwarding in large sensor networks*, Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on, (2001), pp. 304–309.

[155] W. Ye, J. Heidemann, and D. Estrin, *An energy-efficient mac protocol for wireless sensor networks*, INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 3 (2002), pp. 1567–1576 vol.3.

[156] B. Yin, H. Shi, and Y. Shang, *A two-level strategy for topology control in wireless sensor networks*, in ICPADS '05: Proceedings of the 11th International Conference on Parallel and Distributed Systems - Workshops, Washington, DC, USA, 2005, IEEE Computer Society, pp. 358–362.

[157] Y. Yu, R. Govindan, and D. Estrin, *Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks*, 2001.

[158] Y. Zhang and Q. Huang, *Adaptive tree: a learning-based meta-routing strategy for sensor networks*, Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, 1 (8-10 Jan. 2006), pp. 122–126.

[159] Y. Zhang, J. Luo, and H. Hu, eds., *Wireless Mesh Networking: Architectures, Protocols and Standards*, Auerbach Publications, Taylor & Francis Group, New York, first ed., 2006.

[160] J. Zhao and R. Govindan, *Understanding packet delivery performance in dense wireless sensor networks*, in SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, New York, NY, USA, 2003, ACM Press, pp. 1–13.

[161] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, *Impact of radio irregularity on wireless sensor networks*, in MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services, New York, NY, USA, 2004, ACM, pp. 125–138.

[162] M. Zorzi and R. R. Rao, *Geographic random forwarding (geraf) for ad hoc and sensor networks: energy and latency performance*, Mobile Computing, IEEE Transactions on, 2 (Oct.-Dec. 2003), pp. 349–365.

[163] M. Zuniga and B. Krishnamachari, *Analyzing the transitional region in low power wireless links*, Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on, (4-7 Oct. 2004), pp. 517–526.