Adaptive Duty Cycling in Mobile Sensor Networks

Vladimir Dyo

`v.dyo@cs.ucl.ac.uk`

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

University College London

2009

I, Vladimir Dyo, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Mobile wireless sensor networks have recently attracted considerable attention. In particular, there is a significant interest in applying mobile sensor networks to wildlife and environmental monitoring, medical, and human-centric applications. Energy is a critical factor for most real deployments of sensor networks: many mobile applications, such as environmental monitoring ones, require months of unattended operation of large number of small battery-operated nodes. Due to slow advancements in battery and energy harvesting technologies, energy efficiency will remain an important issue for a long time.

The general approach to energy saving in wireless sensor networks is to coordinate the wake-up time of nodes to maximize their sleep time while achieving application goals such as low latency or high throughput. A number of duty-cycling solutions have been proposed for static sensor networks. These solutions often assume a fixed topology and use scheduling techniques to coordinate the wakeup of nodes depending on traffic flows. However, in some applications, a fixed network topology cannot be assumed as some sensors are mobile: duty cycling in mobile networks is challenging because nodes need to continuously scan for neighbours, which is an energy intensive process.

This thesis investigates the issues related to duty cycling of mobile wireless networks. We argue that duty cycling in mobile networks has to be adaptive to both mobility and traffic patterns. The thesis presents a two-level approach which exploits temporal connectivity patterns and offers practical techniques for duty cycling in sparse and dense scenarios.

At macro level, the uncertainty of node discovery is a primary reason of power consumption, which causes the mobile nodes to periodically scan or listen for neighbours, draining battery. At this level, the approach is based on adapting the node discovery procedure to temporal activity patterns inherent to human-centric and animal-centric applications. At micro-level, the uncertainty of packet arrival is a major source of power consumption, and the approach mitigates this by using short-term synchronization, which constrains the packet arrival time to predefined time slots.

The evaluation of the approach is performed through both simulation & implementation

and deployment on a real sensor testbed. In particular, the performance of the macro level protocol is evaluated through simulation with real human and animal mobility traces and through deployment in Wytham Woods (Oxford) for badger tracking. The performance of micro-level protocol is evaluated through measurements on a small scale testbed.

Acknowledgements

I would like to thank my supervisor, Cecilia Mascolo, for providing an opportunity to pursue this PhD, who for almost five years was a constant source of invaluable guidance, support and experience. I feel very privileged to have worked with her and am deeply indebted for the enormous help, support and feedback.

Many thanks to my second supervisor, Stephen Hailes, for his insightful advice and comments in my first and second year viva. I am grateful to Robin Freeman from Microsoft Research Cambridge for opportunity to do internship on Autonomous Habitat Monitoring project, where I learned so much about embedded systems design. I would like to thank Computer Lab, University of Cambridge, for hosting me during the last months of my studies. This work would not have been possible without financial support from Vodafone-EPSRC Dorothy Hodgkin Postgraduate Award.

Thanks to Bence Pasztor, who made numerous trips to Wytham Woods to actually deploy the protocol and collect badger traces. Many thanks to my friends and colleagues from UCL CS: Arman T, Andy, Clovis, Costin, Daniele, Genaina, Jidtima, James, Ilias, Liam, Mohammed, Panu, Petr, Rami, Soo-Hyun, and Torsten. At Computer Lab: Daniele, Salvo and Mirco.

I would like to thank Alla, without her support, I would not have survived the hardships of living abroad for such a long time. Finally, I dedicate this dissertation to my family, who supported me throughout all my life.

Contents

List of Figures

List of Tables

List of Abbreviations

AP	Access Point
ATIM	Ad Hoc Traffic Indication Message
CAP	Contention Access Period
CCA	Clear Channel Assessment
CFP	Contention Free Period
CSMA	Carrier Sense Multiple Access
DCF	Distributed Coordinated Function
DTN	Delay-Tolerant Networking
EWMA	Exponetial Weighted Moving Average
GPIO	General Purpose Input Output
GPS	Global Positioning System
LPL	Low Power Listening
LPM	Low Power Mode
LQI	Link Quality Indicator
MAC	Medium Access Control
MEMS	Micro Electro Mechanical System
MLSE	Minimum Least Squares Estimation
PCF	Point Coordination Function
PIR	Passive Infrared Sensors

PPM	Parts per Million
RFID	Radio Frequency Identification
RLS	Recursive Least Squares
RSSI	Received Signal Strength Indicator
SFD	Synchronizing Frame Delimiter
VHF	Very High Frequency
WoR	Wake on Radio
WSN	Wireless Sensor Networks

Chapter 1

Introduction

Mobile wireless sensor networks (WSN) have received considerable attention in recent years. These networks are starting to be used for wildlife tracking [Juang et al., 2002], livestock monitoring [Sikka et al., 2006, Nadimi et al., 2008], medical [Shnayder et al., 2005], human centric and pervasive applications [Eagle and Pentland, 2006].

Energy is a critical factor in most real deployments of sensor network applications. Many mobile applications, such as environmental monitoring ones, require months of unattended operation of large number of small battery-operated nodes. Due to slow advancements in battery technology, energy efficiency will remain an important issue. Since the introduction of lead-acid cells in 1950s the energy density of these types of batteries have only doubled, with expected rise of only 10-20% in the near future [Kiehne, 2003]. Energy-harvesting solutions are attractive, with photovoltaic, vibrational and thermoelectric techniques being the most promising sources of energy [Coumpis and Aliwell, 2008]. The relatively low efficiency of those devices means energy efficiency will still remain an important factor for a long time.

Since radio communication is a major source of power consumption in wireless sensor network applications [TexasInstruments, 2008], power management often reduces to *duty cycling* the radio, a process where the radio module is turned on and off while making sure the network achieves its aims. Many duty cycling approaches exist for *fixed* wireless sensor networks, for instance [Chen et al., 2001, Polastre et al., 2004, Ye et al., 2004, Hohlt et al., 2004, Lu et al., 2004, Rhee et al., 2005, Keshavarzian et al., 2006]. The majority of these focus on energy efficient channel access and arbitration with the general goal of obtaining an optimal trade-off between latency and power consumption. The approaches can be grouped into synchronous and asynchronous categories. Synchronous approaches [Lu et al., 2004, Rhee et al., 2005, Keshavarzian et al., 2006] assume global time synchronization either with precise clocks or by periodically resynchronizing through network time synchronization protocols [Elson et al., 2002]. Asynchronous approaches [Polastre et al., 2004, El-Hoiydi and Decotignie, 2004] do not assume the presence of precise clocks, and the nodes duty cycle asynchronously and independently from each other.

However, current approaches are not applicable to *mobile* wireless sensor networks, which are characterized by intermittent connectivity of the nodes. In addition, in *sparse* mobile networks, the nodes can be isolated but whenever a new node comes into communication range it has to be detected in timely manner in order to initiate communication. Thus, mobile applications cannot be simply built on top of a power efficient MAC layer designed for fixed networks. These MAC protocols usually perform a neighbour discovery only once during configuration phase, and then schedule the nodes to be active during certain timeslots and sleep at any other time. This leads to network partitioning and failure to timely detect new neighbouring nodes [Drula et al., 2007], and consequent inability of a networking protocol to deliver messages.

The crux of the problem is that the nodes have to be discovering neighbours while spending the minimum amount of time in active mode, in mobile networks this is excercebated by the dynamics of the nodes. Wireless routing protocols such as DSR [Johnson and Maltz, 1996], AODV [Perkins and Royer, 1997], OSPF [Moy, 1998] [Vahdat and Becker, 2000] [Lindgren et al., 2003] rely on periodically sending broadcast packets to advertise the node's presence and discover other nodes. This assumes all the nodes have to be listening most of the time to be able to timely discover other nodes or be discovered by other nodes. This is problematic for modern radios where the cost of idle listening is approximately comparable to the cost of transmitting or receiving [TexasInstruments, 2008]. In the worst case, a node would have to be awake all the time, waiting for a contact with other nodes, which would limit the useful lifetime of a typical node to just 30 days[1].

There have been a number of recent attempts to apply duty cycling to mobile wireless networks such as [McGlynn and Borbash, 2001], [Zheng et al., 2003], [Jiang et al., 2005], [Jun et al., 2006], [Borbash et al., 2007], [Dutta et al., 2008]. Most of these approaches focus on node discovery functionality, but do not deal with channel access and duty cycling in connected states. Some of these protocols have been evaluated in a simulator with a number of assumptions about the hardware. The approaches are based on generating a certain sequence of states between listening, transmitting and idle states, which are designed to maximize the probability of overlapping between the active states of asynchronously duty cycled nodes. [McGlynn and Borbash, 2001] looks at energy efficient node discovery after sensor nodes are deployed in the field. [Zheng et al., 2003], [Jiang et al., 2005] and more recently [Borbash et al., 2007] investigate an efficient way for mobile nodes to discover neighbours using birthday protocols. [Dutta

[1]TMote Sky 30mA @3V, with lithium D size battery, 19Ah, operating at room temperature.

et al., 2008] presents an asynchronous protocol for node discovery based on prime numbers.

Surprisingly, there is little research on whether it is possible to exploit patterns in network activity and use these patterns for duty cycling, specifically for energy-efficient node discovery and communication. Recent research has shown that the mobility of humans [Eagle and Pentland, 2009] and animals [Ebert et al., 1983] is non-random and exhibits patterns, which have successfully been exploited for routing, location tracking and context-aware applications. The spatial patterns have been addressed in the context of probabilistic routing [Lindgren et al., 2003][Musolesi et al., 2005][Chakrabarti, 2003], location tracking and context aware systems [Matsuo et al., 2007]. Probabilistic routing protocols [Lindgren et al., 2003] [Musolesi et al., 2005] [Spyropoulos et al., 2005] exploit the fact that different pairs of nodes have different encounter probabilities. This information is used to make intelligent decisions on message forwarding and message replication. Location tracking algorithms and context aware systems [Matsuo et al., 2007] extract patterns from user mobility to infer high-level user behaviour. These patterns, however, have not been studied in the context of duty cycling.

To summarize, this thesis focuses on the problem of duty cycling in mobile wireless sensor networks. The nodes in these networks are faced with contradictory goals. There is a need to timely discover the neighbours and, at the same time, to maximize the sleep time of all nodes. We explore the potential of exploiting encounter patterns in wireless sensor networks and using those patterns for better power management. We investigate the techniques for both energy efficient node discovery using patterns, as well as, MAC layer techniques for communication and channel access.

1.1 Assumptions

In this thesis we make the following assumptions:

- The detection of neighbours only happens through normal short-range radio. Node detection is not a problem if nodes are equipped with specialized sensors. However, these devices increase the cost and the size of the equipment and are not always available or deployable (e.g., in our wildlife monitoring application, which needs cheap and very small sensor nodes).

- Need for very low energy consumption. This is very important for applications where battery replacement and energy scavenging cannot be used or can only be used sparingly. The amount of energy the nodes can carry or generate is limited by the size of the tag, which in our scenario has to be small to minimize the strain on the animal. Replacing the batteries is not desirable, as recapturing is a very labour intensive and intrusive process.

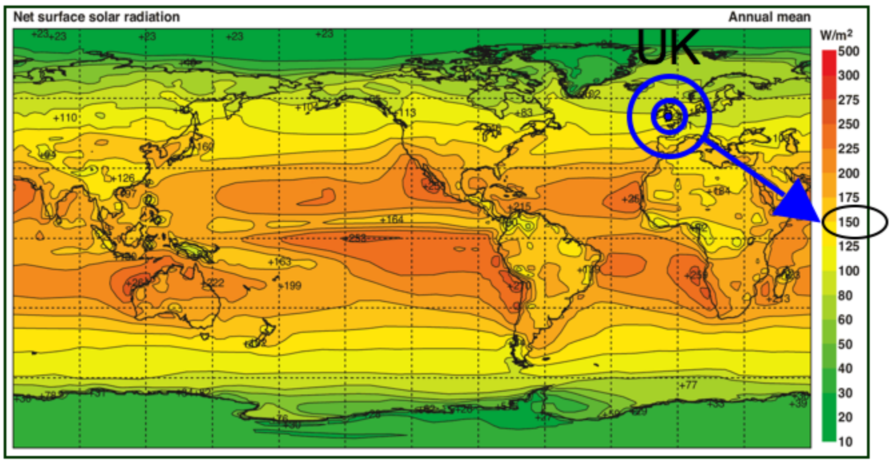


Figure 1.1: A diagram of net surface solar radiation, annual mean.

Energy scavenging techniques are very attractive long-term solutions for powering mobile devices. The efficiency of such technology is still low (Table 1.1) and the total amount of energy is limited by the small footprint, size and cost of the device [Chalasani and Conrad, 2008].

The use of solar panels, one of the most promising sources of energy, is possible but currently limited by low efficiency of panels and daily activity of animals. For example, it cannot be used on nocturnal animals[2].

- Time synchronization. The low cost nodes have a clock drift, which depends on temperature, voltage and varies from node to node. Precise time synchronization is possible with high precision local clocks, GPS or AM/FM radio time synchronization, but would increase the cost of the tag, which is an important factor for larger deployments. The use of time synchronization protocols [Maroti et al., 2004] is of course possible, but is limited to connected areas of the network only.

  We assume the presence of local clocks with a limited precision. The clock precision is limited by the cost of a crystal and is about 50 ppm (part per million) for typical low cost sensor nodes.

- Mobility. The tags will be attached to animals and some of the tags might be additionally deployed statically in the habitat. The study considers social animals, and we assume that the tags will frequently come into contact with each other. However, since the animals can have large habitat area, the network is likely to be intermittently connected. The

[2]Grey seals can come to the shore at night times only, whereas a short surfacing interval is not enough to provide enough energy. Badgers lead a nocturnal life sleeping during the day.

Energy source	**Efficiency**	**Notes**
Battery	1.12kJ/cm^3 (Alkaline), 0.82kJ/cm^3 (Lithium-ion)	1mW application operating over 2 years would require 54cm^3 of alkaline battery.
Photovoltaic	10mW/cm^2, @12% efficiency in direct sunlight.	Varies significantly with time of day, climate and location. For example, the annual average solar radiation in the UK is $15mW/cm^2$ (Figure 1.1), which takes into account daylight duration, seasons and weather conditions. Cannot be used for nocturnal animals such as badgers.
Thermoeletric	1.5-40μW at temperature gradient from human body to ambient environment, radioactive thermoeletric cells - 370μW	Delivering power from generator to the device involves wires.
Electromagnetic	Depends on range	Typical operating range is 1-2m for inductive coupling, and $<$10m for backscattering.
Vibration	0.05-9mW	The drawback is large size and weight, 25-300g for given power levels.

Table 1.1: Energy harvesting characteristics.

protocol should therefore provide mechanisms for energy-efficient neighbour discovery as well as energy-efficient communication in sparse and dense scenarios.

- Support for modern packet-based radios such as CC2420 [TexasInstruments, 2008]. The cost of operating these modern radios is the same in transmitting, receiving or idle listening states. Thus the network energy efficiency is determined not only by the amount of traffic but by the amount of time the radio is switched on. We assume that the same radio is used for communication and proximity detection.

- Support for interactive (quering) and replication (buffer synchronization) traffic, where nodes have to frequently synchronize the contents of their buffers, and run interactive query sessions. For example, a node might need to periodically send queries or report changes of its buffer to its neighbour.

- Mobility is not a source of energy consumptions: this is true for applications involving animals and humans. We further assume that wireless sensor operations do not have any impact of animal behaviour, i.e. node mobility.

Throughout the thesis, the terms *mobile node* and *tag* are used interchangeably. We will also use the terms *duty cycling* and *scheduling* interchangeably in this thesis.

1.2 Research Problem and Thesis Contributions

In this thesis we investigate an integrated approach for energy efficient node discovery in mobile wireless sensor networks. We look at the problem in a holistic way and provide an integrated two-level approach to save power in mobile sensor networks. These two levels have access to different type of information and can make decisions at different levels of granularity. The micro-level deals with packet transmissions, advertising beacons at a certain rate and actual discovery of neighbouring nodes. At this level we try to minimize idle listening time, overhearing and transmission costs. While the macro-level may 'learn' or 'know' when the contacts are not expected and completely switch off the nodes during that period. We evaluate both approaches using a combination of implementation on Tmote Sky sensor nodes, simulations, measurements on a testbed and a real deployment.

1.2.1 Micro-level Scheduling

At a micro-level, we present MAC*ron*, a lightweight duty cycling protocol, which combines the features from both asynchronous and synchronous MAC protocols.

The protocol builds upon a low power listening technique [Polastre et al., 2004, El-Hoiydi and Decotignie, 2004] for wake-up scheduling in mobile wireless sensor networks. In low power listening, the nodes periodically sample the radio for energy and go back to sleep if there is no activity. When a node needs to send a packet, it prepends this packet with a long extended preamble to wake up nearby nodes. As low power listening puts a large overhead on the sender for transmitting each packet, the protocols that are using it [Polastre et al., 2004, El-Hoiydi and Decotignie, 2004] can be efficient for low throughput situations but suffer in dense or high-throughput scenarios.

The protocol allows to achieve ultra low duty cycling level of 0.3% while providing low latency in sparse and dense scenarios. It learns the relative clock drifts between any pairs of nodes and uses this information to minimize the transmission costs of the nodes. The protocol has a novel schedule exchange extension, which further reduces the power consumption in dense scenarios. The schedule exchange extension helps multiple nodes in the network to coodinate their schedules without using expensive broadcast transmissions. Finally, the protocol introduces a novel *clock quality metric*, which minimizes the clock aberration while transferring timestamp information through nodes with drifting clocks.

1.2.2 Macro-level Scheduling

At a macro level, we propose an energy efficient node discovery protocol, which detects and exploits temporal regularities in node arrivals.

Existing research results from other disciplines, such as biology or sociology has shown that the mobility of individual animals is not random and exhibits some seasonal, daily and other patterns. The physiological processes of living beings, including plants, animals and bacteria follow a 24-hour cycle, called a circadian rhythm. These temporal cycles depend on social structure, availability of food, migration and other factors and have been extensively studied in chronobiology [Ebert et al., 1983].

We designed a learning algorithm and apply it to partially mobile [Dyo and Mascolo, 2007] and fully mobile wireless sensor networks [Dyo and Mascolo, 2008]. In a partially mobile sensor networks, a sink (or infostation) located in a strategic position needs to detect and communicate with mobile sensors. The algorithm we developed aims at maximizing the number of mobile node detections given sink's daily energy budget. The algorithm uses a simple reinforcement learning technique [Kaelbling et al., 1996] to learn the mobile node arrival patterns and uses this knowledge to drive the sink's duty cycle. In fully mobile wireless networks, the mobile nodes can use the algorithm to increase the number of encounters between each other, given a fixed energy budget.

1.2.3 Evaluation

To demonstrate the benefit of our approach we have implemented both protocols in TinyOS [Levis, 2006] for Tmote Sky sensor nodes [MoteIV, 2006]. We evaluate the MAC*ron* protocol by measuring the power consumption on a small-scale testbed. We show the performance gains achieved by skew correction and schedule exchange features of our protocol for different network densities under various traffic loads.

To evaluate the effect of the macro-level duty cycling we used five indepedent real human and animal mobility traces. We used human mobility traces from MIT Reality Mining [Eagle and Pentland, 2006] and Intel Haggle [Scott et al., 2006] project, collected by attaching Bluetooth devices to a number of volunteers over a long period of time. In the evaluation, the algorithm was also tested against synthetic traces with variable parameters, and on animal traces, collected by attaching small GPS tags to brushtail possums for over a year [Dennis et al., 2008]. Assuming those traces represent a ground-truth data about node movements, we use those traces to drive the node movements in a network simulator. A lightweight and efficient learning mechanism should be able to detect temporal pattern in node arrivals, for example, a nocturnal pattern and put nodes to sleep more often during the daytime, when there is little activity. This should result in reduction of energy consumption while detecting the same number of encounters or delivering the same number of messages in the network. We identify the conditions when our approach is beneficial as well as its limitations.

Finally, the macro-level approach has been evaluated through a real deployment to track the population of badgers in Wytham Woods, Oxford, UK as part of the WildSensing project [Dyo et al., 2009]. The approach has been used to extend the system lifetime by adaptively controlling the duty cycling of energy consuming RFID readers. To evaluate the efficiency of our approach, the system running an adaptive algorithm was deployed in parallel with always-on readers and shown to detect higher number of encounters per unit of energy than random duty cycling.

1.3 Thesis Outline

The thesis contains 8 chapters. Chapter 2 contains a motivating scenario and background information. Chapters 3 and 4 present the micro and macro level scheduling protocols. Each of these chapters is followed by a relevant work section. Chapters 5, 6 and 7 present an evaluation and discussion of the results. Chapter 8 concludes the thesis and describes potential future work. The chapters are organized as follows:

- In Chapter 3 we present the design of MAC*ron* protocol with mobility support. We first motivate the need for energy-efficient neighbour discovery, neighbour management and communication. We then propose a protocol that uses novel skew estimation and schedule exchange, novel techniques to improve power efficiency of mobile applications.

- In Chapter 4 we present the macro-level scheduling protocol, which controls the parameters of the MAC protocol described in Chapter 3. We first present a scenario for single hop networks, where a static sink needs to detect and communicate with mobile sensors. The protocol, we designed for this scenario, uses basic techniques from machine learning to detect encounter patterns and use them schedule sink's energy budget. We then extend an approach for fully mobile multi-hop networks, where each node in the network is adapting its duty cycle depending on local information about encounters.

- In Chapter 5 we present an evaluation of the micro-level protocol. We describe the protocol implementation on Tmote Sky sensor nodes and present experiment results conducted on a small scale testbed. The performance is evaluated by measuring the network power consumption for different node densities, traffic rates and duty cycles. We discuss the results and compare it with other approaches.

- In Chapter 6 we present an evaluation of the macro-level protocol. We evaluate the protocol on both synthetic and existing human mobility traces in Tossim simulator. We discuss the choice of parameters and trade-offs between power efficiency and performance metrics such as delivery ratio or number of detected encounters.

- In Chapter 7 we present an implementation, hardware platform and deployment results of macro-level approach for active RFID badger tracking application in Wytham Woods, Oxford, UK.
- In Chapter 8 we conclude the thesis and outline possible future work.

Chapter 2

Background and Motivation

2.1 Motivation

In this section we describe a scenario, which provides a practical motivation for the methods developed in this research. In the scenario the mobile wireless sensors are attached to animals (Figure 2.1) to collect information about individual mammals such as temperature or encounters between the animals over a long period of time. This information will help conservation biologists to understand the feeding habits, the social and behavioural patterns of species.

Potential target species are badgers (*meles meles*), a controversial animal in the UK, which is believed to be responsible for the spread of tuberculosis among cattle [Donnelly et al., 2006], making it very important species to be studied in terms of ecology and behaviour. Grey seal (*Halichoerus grypus*), the most populous mammal in the UK territory, is another potential species, presenting a significant interest to conservation biologists [Thompson et al., 1991]. Both species have been studied before using conventional telemetry solutions, which are either intrusive and labour intensive (VHF beacons) or very expensive (GPS loggers) [Mech and Barber, 2002]. We briefly describe these methods below.

The conventional radio telemetry systems have been successfully used for analysis of movement, desease transmission, breeding behaviour and many other biological applications. The early systems consisted of a VHF transmitter periodically emitting a radio beacon on a unique frequency. Tracking a single animal was done by several researchers each carrying a mobile VHF receiver tuned to the tag's frequency and manually triangulating the animal position by VHF beacon signal strength, an intrusive and extremely labour intensive method. More advanced GPS collars with Satellite/GSM transmitters allow fully automated tracking but are still very expensive, which limits the scale of current deployment to just a few animals [Mech and Barber, 2002] and have some further limitations, such as cannot operate under deep foliage and high power-consumption.

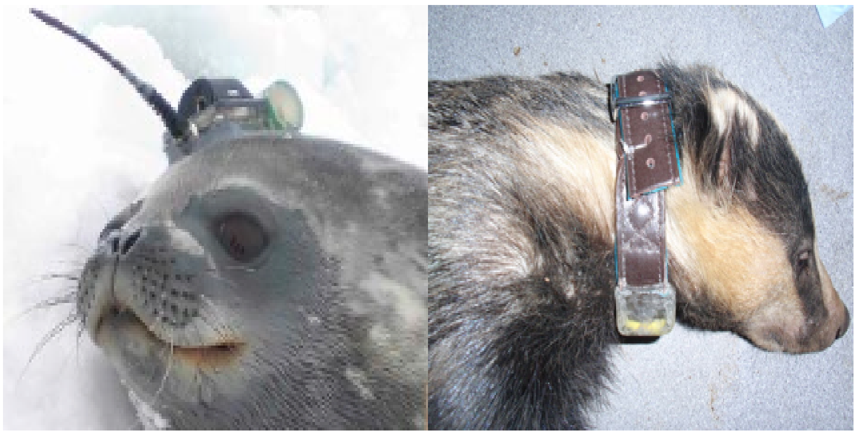

(a) GPS tag on a grey seal. (b) Active RFID tag on badger.

Figure 2.1: A picture of a GPS/satellite tagged seal and an RFID tagged badger. Photo courtesy of Sea Mammal Research Unit at St. Andrews and Department of Zoology, University of Oxford.

The scenario in this thesis considers a novel method, where each animal is tagged with a low-cost wireless sensor. Compared to VHF transmitters, the wireless sensors would enable unintrusive monitoring and tracking of individual animals. Compared to satellite-based solutions, the low-cost wireless sensors would enable to conduct experiments on a much larger scale. This would allow to study complex interactions between the species and identify the behavioural patterns.

The intended operation of the method is as follows (Fig. 2.2). Each individual sensor is periodically logging some information, such as temperature or contact and stores it in a local storage. As the animal population can span geographically large areas, the wireless nodes form a *sparse* mobile network and rely on opportunistic routing protocols [Vahdat and Becker, 2000] to deliver data to the base station. Whenever two animals (same specie) get in communication range, the wireless nodes transmit the contents of their buffers to each other, the messages eventually percolate through this pair-wise exchange to one of sink nodes, which sends it to a base station through long-range communication links (such as GSM).

The opportunistic routing protocols have been studied extensively in recent years in terms of delivery ratio, message overhead, buffer sizes, and the node mobility [Vahdat and Becker, 2000]. The opportunistic protocols have been shown to achieve high delivery rates with limited buffer sizes, and the main thrust of research has been on reducing the message overhead while maintaining high delivery rates [Lindgren et al., 2003], resulting in a family of protocols using a single or a few message replicas to achieve a high message delivery rate [Small and Haas,

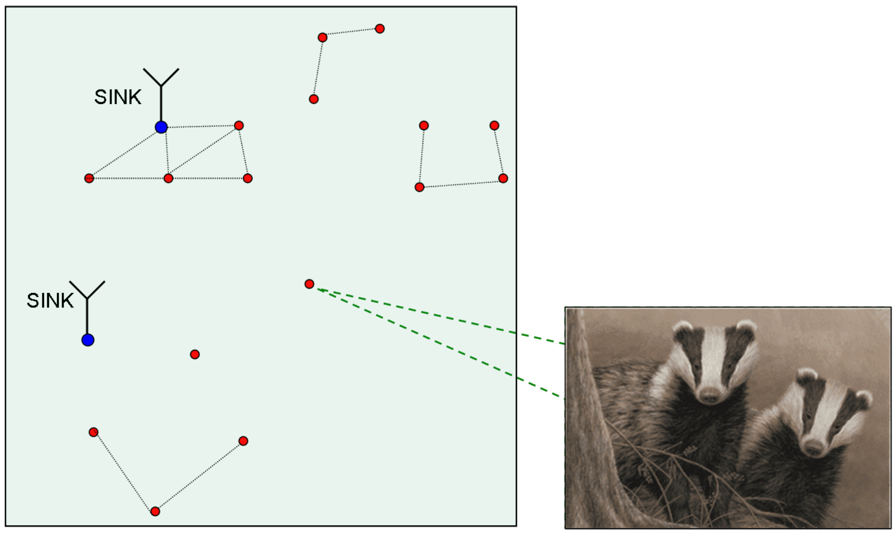


Figure 2.2: Scenario.

2003] [Musolesi et al., 2005] [Spyropoulos et al., 2005].

2.2 Background

The central problem in the described scenario is an ability of wireless sensor nodes to timely discover each other in an energy-efficient way. Timely discovery is essential for maintaining network connectivity and efficient data collection from the wireless tags attached to animals.

In this section we describe existing approaches to node detection problem starting from a neighbour discovery problem formulation, device discovery in modern protocols and hardware-oriented detection techniques, such as wake-up circuits, accelerometers and passive infra-red sensors.

2.2.1 Neighbour Discovery Problem

A neighbour discovery is related to *rendezvous problem* originally formulated by [Alpern, 1995]. In this problem two people need to find each other in a large maze with n rooms without agreeing on a meeting place beforehand. At each timestep, each person can either stay in a current room, or move to any other room. The goal is finding an optimal room sequence to minimize the search time. The *rendezvous problem* is widespread in nature, for example, when non-social animals need to meet during a mating season. It also has applications in the areas of operating systems, synchronizations, robotics [Roy and Dudek, 2001] and mobile agents [Kranakis et al., 2006]. The basic solution to this problem is a randomized strategy, when an agent either stays in a current room with a certain probability or moves to a random room. An optimal strategy has been found for $n <= 3$ [Weber, 2009]. For very large values of

n, the expected time to meet with a randomized strategy is $0.8289n$ steps, although the optimal solution is still unknown.

The node discovery problem can be mapped to rendezvous problem: the unsynchronized duty cycling nodes wake up at different time slots and need to find an optimal sequence of active timeslots to reach a rendezvous point when both nodes are awake. Depending on the design of this sequence, there are several approaches. [Zheng et al., 2003] [Zheng et al., 2006] propose a distributed asynchronous wake-up protocol for mobile networks using block design problem from combinatorics. The nodes in the protocol follow a pre-calculated on/off sequence, which leads to faster discovery times. The algorithm is deterministic and there is a probability that a set of nodes will systematically interfere with each other. [Jiang et al., 2005] calculate a wake-up sequence using quorum-based algorithms. The protocol is based on the specially designed non-random wake-up sequence, so that any phase shift of this sequence will still overlap with each other resulting in nodes discovering each other. As a result, duty cycled nodes are able to detect each other's presence with a low latency. [Borbash et al., 2007] calculate the optimal sequence by using birthday protocols. The protocol is designed to work during the deployment stages, when the nodes need to discover each other. [Dutta and Culler, 2008] recently proposed an asynchronous duty cycling protocol for mobile node discovery using prime number theory. In the protocol, each node has two prime numbers and a counter. Whenever counter is equal to one of the numbers, it turns on the radio and listens for packets from potential neighbours.

The approaches described above can provide node discovery in fewer than n steps, however, as the nodes have to wake up at a certain sequence, which can be calculated online or preprogrammed, the approach results in a higher complexity. The other potential problem is that the randomized sequence can interfere or complicate the ability of nodes to synchronize their time slots to reduce the communication costs or latency.

The other group of approaches use simpler discovery mechanism, where the node periodically wakes up for an entire duration of n slots to either continuously listen for beacons or continuously transmitting (low power listening). [Ye et al., 2004] uses periodic scans during which a node stays awake for extended periods of time and listens for beacons from other nodes. The scan frequency is inversely proportional to node density, so the node would scan more frequently when it is isolated and less frequently if it is surrounded by many neighbours. The assumption is that the network normally operates in connected mode, with each node having constant connectivity to a known number of neighbours, which is not the case for mobile networks. [Lin, 2005] proposes transmitter initiated (TICER) and receiver initiated (RICER) asynchronous duty cycled schemes for wireless sensor networks. In both schemes, the nodes are duty cycling asynchronously from each other. In transmitter initiated scheme, a transmitter uses a packet train to wake up a receiver node. In the receiver initiated scheme, the nodes periodically transmit short beacons, and enter into relatively long idle listening state, when they are ready to receive data. In this listening state, the node responds to any short beacons from other nodes it hears and initiates a data transmission. The proposed methods were designed for static networks and do not consider mobility.

At a higher abstraction layer, there are clustering approaches targeting spatial redundancy in dense scenarios. The problem of distributed power control has been considered in the SPAN protocol [Chen et al., 2001]. SPAN uses a distributed master eligibility rule where each node determines whether it should become a master. The rule is that if two of the node's neighbours cannot reach each other either directly or via one or two masters, it should become a master. The master nodes are periodically shuffled to avoid burning out master nodes. The protocol is robust and provides significant energy savings. [Xu et al., 2001] uses geographical information to split the network into a virtual grid and the node with the highest residual energy becomes a cluster head. The rest of the nodes in the grid can go to sleep. Each node can be in either sleeping, discovering or active states. A node starts in discovery state and becomes a master if it has not heard from any other node. The master node is active for a certain time and then changes its state to discovery giving other nodes opportunity to become masters. In PEN [Girling et al., 2000] the nodes use asynchronous duty cycle. Each node periodically wakes up, transmits a beacon, listens for replies and goes back to sleep.

2.2.2 Device Discovery in Current Wireless Standards

2.2.2.1 Bluetooth

Bluetooth is a wireless communication developed for data communication with mobile devices over short distances. A number of Bluetooth nodes form a Personal Area Network (PAN) or a PicoNet. Bluetooth nodes use special discovery procedure for discovering each other. For discovery purposes, a device (master) periodically enters an *inquiry* state, where it sends a beacon and listens for response. The other device (slave) periodically enters an *inquiry scan* state, where it listens for and responds to beacons.

Bluetooth nodes use frequency hopping over 79 frequencies. The hopping sequence is pseudo-random, so the devices cannot communicate unless they discover each other. The devices use 32 frequencies out of 79 for discovery purposes. In the inquiry state, the master device hops through a specially generated sequence of 32 frequencies sending beacons and listening for response in the following way. At each timeslot the master node sends 2 beacons at 2 different frequencies. At odd timeslots, the master node listens for response. The beacon contains a device address and local clock information, so that listening acquire the system time and follow the hopping sequence. The slave device periodically listen for beacons for short period of time and responds to a beacon. The default duration of inquiry procedure is 10.24s and 5.12s for Bluetooth v1.1 and v1.2 respectively.

The problem of neighbour discovery in Bluetooth networks has been addressed by [Kuijpers et al., 2002]. The authors observe that network layer relies on broadcast capability of a link layer. Bluetooth does not have broadcast capability and emulates broadcast by multiple unicast frames. The authors propose to replace emulated broadcasting with some mechanisms from Bluetooth profiles.

2.2.2.2 IEEE 802.15.4

802.15.4 is an emerging wireless standard for low power low data rate WLANs. The standard supports star and peer to peer topologies, and specifies beacon and non-beacon operation modes. In beacon mode, the coordinators periodically transmit beacons. A beacon is followed by Contention Access Period (CAP) and then by Contention Free Period (CFP). The nodes use CAP for signalling purposes to reserve timeslots within the Contention Free Period, which is used for actual data transmission.

Zigbee specifies a *device discovery* application primitive, which queries devices in the active channel. In 802.15.4 compatible networks, the scanning is done using extended preamble. Because the maximum preamble length is limited by the 802.15.4 to 32 bits, the extended preamble is emulated by transmitting short packets back to back as frequently as possible. The nodes periodically perform carrier sampling and sleep at other times. The extended preamble is not limited in length, and its optimal duration depends on the beaconing frequency. The higher beacon frequency ensures timely node discovery, but generates large amounts of traffic. For example, transmitting 1s preamble every minute is equivalent to sending of 45Mb of traffic per day[1].

2.2.2.3 IEEE 802.11

IEEE 802.11 is a standard for Wireless Local Area Network communications. The standard uses CSMA/CA protocol and two mode of operations: Distributed Coordinated Function (DCF) for infrastructure and Point Coordination Function (PCF) for ad hoc mode. The infrastructure (DCF) mode specifies a power save mode, where radio is active only a fraction of time. In this mode an Access Point (AP) buffers the incoming packets for its registered client stations, and periodically broadcasts beacons, called Ad Hoc Traffic Indication Message (ATIM), which

[1]Raw traffic based on 802.11 250kpbs data rate.

indicates whether an AP has any buffered packets for each station. The clients periodically wake up to receive a beacon message, and request data transfer, if needed.

This mechanism requires the clients to be synchronized with the access point and a static non-changing topology. The frequency of beaconing, the node discovery is not addressed by this protocol. Some mechanisms for duty cycling in 802.11 networks have been proposed [Jiang et al., 2005] but evaluated in simulations only.

2.2.2.4 Passive and Active RFID

The RFID technology is specified by ISO/IEC 18000 set of standards [International Organization for Standardization, 2008] containing the architecture, addressing, communication and power requirements for tags and readers. Depending on the presence of battery on the tag, there are three types of RFID devices: passive, active and semi-active.

Passive RFID require no power source and can operate without batteries. The tags are discovered and activated by a powerful radio impulse from the reader. The passive devices are based either on induction coupling or backscattering principle [Roussos, 2008]. Inductively coupled tags have to be located within the electromagnetic near field, where the electromagnetic field is being generated. The tags respond to the reader by modulating the current flowing through the circuit, which is detected by the reader. The backscattering devices operate in the far field, where mutual effect of antennas is minimal, the devices collect energy from the reader and use it to power CPU and transmitter. To discover neighbours, the reader has to continuously transmit radio impulses. The radiating power of a reader is subject to regulatory limitations and is typically 1W[2], which limits the operating range of the tags to several meters.

Active RFID operate on a battery and have longer operating range compared to passive tags. Due to the battery or other energy source, these tags have larger size, higher cost and limited lifetime. The communication costs in active RFID technology is shifted towards the readers, which are powered on all the time, and the tags only wake up to transmit the beacon. The tags can operate for relatively long periods of time, typically more than a year on a single battery.

Semi-active RFID tags (also called semi-active) are battery assisted and combine the advantages of passive and active RFID technologies, by using passive RFID circuit to detect the transmission and active, battery powered circuit to transmit the response to the reader.

2.2.2.5 Ultra-low Power Wake-up Circuits

There are some approaches [Gu and Stankovic, 2005], [Pletcher, 2008], which use a separate very low power radio receiver to monitor the channel while the node is asleep. The separate

[2]In the US and Europe.

radio is called a wake-up circuit and generates an interrupt when it detects energy in the channel. The node then uses a main radio for communication. A recently developed PicoRadio prototype [Pletcher, 2008] uses a novel "uncertain-IF" architecture providing -72dBm sensitivity at 2Ghz and consumes 52 μA. The approach developed by [Gu and Stankovic, 2005] uses a separate wake-up circuit, which can be remotely powered by a transmitter by transmitting at a certain frequency. This operation relies on a relatively powerful transmitter to send an wakeup impulse to all nodes.

The advantage of wake-up circuits is that they have a potential to provide both low latency and very low power consumption. As the wake-up circuit monitors the channel continuously, the node can be woken up almost immediately, as opposed to duty cycling approaches [Rabaey et al., 2000].

A potential limitation is a different communication range for the main and wake-up radios. The detection range provided by such wake-up circuit is much smaller than the communication range of the main radio. The operating range of the wake-up circuit designed by [Gu and Stankovic, 2005] is 7m, which is much smaller than normal operating range of 802.15.4 radios used for communication. To minimize the power consumption the wake-up circuits have to use simple modulation schemes, such as on-off key modulation, which are inherently more susceptible to noise than DSSS used in 802.15.4 radios. This would translate to more powerful transmitters or shorter communication ranges for the same bit rate compared with direct spread spectrum radios. Finally, ultra-low power wake-up circuits are not yet widely available, and would increase the cost of the nodes, an important factor for large deployments.

The new generation of byte-level radios, such as CC11xx and CC25xx from Texas Instruments [Instruments, 2008] have a very low power consumption in active and standby modes. These radios have a mechanism called Wake-on-Radio functionality (WoR), which is a firmware implementation of low power listening functionality. This means that periodic clear channel assessment (CCA) checks are performed by the radio chip without involvement of a microcontroller. Although more efficient than software implementation, WoR inherits all the limitations of low power listening based protocols, such as the high wake-up and transmission costs and high overhead in dense environments.

2.2.3 Adaptive Scanning

Adaptive Scanning for Node Discovery. The need for adaptive scanning has been recognized in recent works on practical deployment of mobile wireless system [Su et al., 2006] and mobile node device discovery [Drula et al., 2007]. [Su et al., 2006] proposes a *variable inquiry rate* to collect Bluetooth mobility traces. To save power the nodes were configured to sample the neighbourhood more frequently when no nodes are detected and then reduce the sampling rate if there are nodes around. Although an approach was used to actually collect traces, there was no evaluation quantifying the impact of this technique on the number of detected encounters. [Drula et al., 2007] recently proposed an approach for adaptive scanning in Bluetooth networks. The authors propose two algorithms for dynamically adjusting the Bluetooth parameters based on past activity in ad-hoc network. The algorithm was evaluated using simulations and showed to reduce energy consumption by up to 50% compared to static power-conserving scheme. The authors used random waypoint mobility model augmented with attraction areas - areas towards which the nodes move with high probability.

Adaptive Scanning for Data Collection. The idea of adaptive sampling as opposed to sampling at fixed intervals is currently being researched in the areas of autonomous ocean sampling, environmental sensing and environmental robotics. In conventional sampling, the sampling interval and parameters are fixed prior to the survey and usually rely on knowledge of the phenomena. In most of these applications, sampling at regular fixed intervals is either not efficient or not possible. Adaptive sampling allow for adapting the sampling frequency online, depending on the past observations. For example, when sampling animal population, the sampling can start at low resolution and, when a certain species is detected, increase resolution for all neighbouring areas. As opposed to standard randomized sampling, an adaptive sampling mechanism could direct limited resources into the most interesting parts of the phenomena thus gaining most information maximum information content per unit of energy.

Adaptive sampling approaches have been studied for climate prediction systems [cas], environmental monitoring [Krause and Guestrin, 2007] and clinical trials of new drugs [ada]. In the latter area, for example, the goal is to compare the impact of several drugs on people. The standard procedure assigns equal number of people to each drug for an entire experiment, which leads to simple comparison between the drugs, but not efficient when the drugs show different performance early on in the experiment. An adaptive strategy dynamically reallocates people in the groups to better performing drugs, thus increasing the benefit of the drug and reducing the time it takes to make a decision.

[Thompson and Seber, 1996] contains a survey of adaptive sampling designs for environmental applications, such as rare animal and plant species detection. [Eickstedt, 2006] [asa] researches the use of adaptive sampling for oceonographic monitoring using a group of autonomous underwater vehicles. [Willett et al., 2004] applies the concept of adaptive sampling to distributed data collection in wireless sensor network. The approach, called Backcasting, is developed for sensing spatially correlated events. It is based on collecting information from

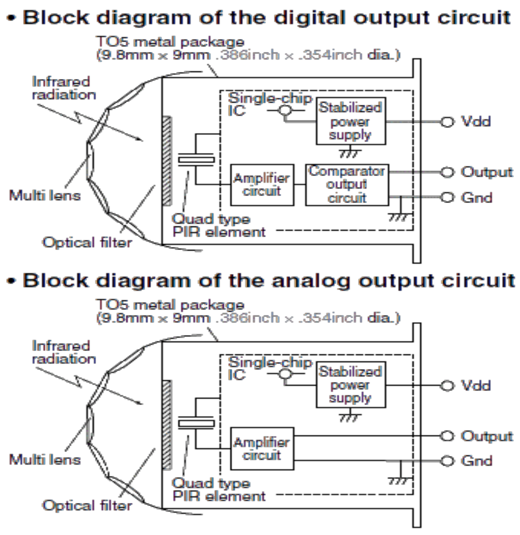


Figure 2.3: Panasonic NapiON passive infrared receiver.

a small subset of sensors and then selectively activating additional sensors to achive a target levels of error. The approach has been analyzed theoretically and shown to reduce power consumption by a factor of 10 while providing the same accuracy as a sensor network, which is activated all the time. [Rahimi et al., 2004] uses adaptive sampling for environmental robotics. The work develops an adaptive method for balancing the quality of sampled data with energy costs. The robotic agent, first explores the phenomena using a nested stratified sampling approach and then iteratively increases the sampling resolution in an online fashion depending on the previous measurements results. The approach allows for systematics reconstruction of the phenomena from the data collected using the proposed adaptive sampling approach. Finally, an adaptive sampling has been used to adjust the sampling rate depending on the predictability of the phenomena [Chatterjea and Havinga, 2008]. The incoming data is modeled using time series models and the data rate depends on the residual error between the predicted model and the actual measurements. None of the works, however, evaluated the adaptive sampling for exploiting temporal correlations for node discovery in mobile environments.

2.2.4 Hardware Detection Mechanisms

The other methods for detecting mobility include: passive infrared sensors, motion detectors and accelerometers. We will compare them with the respect to size, specified performance and power consumption.

Passive infrared (PIR) sensors detect the changes in infrared radiation, which occur when there is a movement by a person. PIR sensors can detect movement within typically several meters within a certain angle (80-100 degrees) and have low power consumption. The sensor is equipped with a set of lenses and an optical filter to focus IR radiation on PIR element. They are still relatively large, for example, the Panasonic 'Napion' MP motion sensor [Panasonic, 2009], marketed as the world's smallest with a built-in amplifier has a cylindric shape 9.8mm long and 9mm in diameter (Figure 2.3). To sum up, these devices are suitable for stationary and batery equipped systems, where the size is not critical.

The accelerometers can help detect the motion of the object carrying the sensor itself. The modern low power accelerometers can detect such events as motion, speed in 3 axis, free fall, and inclination. They are more compact than PIR sensors, and do not require lenses, but cannot detect the movement of other objects. To use for mobile networking, this would require moving objects to continuously advertise their presence, with a frequency depending on the speed. The typical power consumption of low power 3-axis accelerometers is 1mA in active and 1-10μA in standby modes (STM LIS3LV02DQ MEMS inertial sensor [STMicroelectronics, 2005]).

2.3 Summary

In this section we have described existing methods for the device discovery in the wireless networks. The problem can be modeled theoretically as a *randezvouz* problem. The solution to this problem is found by designing a special sequence of active and sleep states, which guarantees discovery of nodes in a certain time. The randomized methods, which select the states of the timeslot randomly, allow nodes to discover each other in $0.8289n$ steps, although the optimal solution to this problem has not been discovered so far. Existing duty cycling protocols in this area, address the problem of neighbour discovery, but do not consider communication and channel access.

We have then provided an overview of device discovery in modern wireless protocols, such as Bluetooth and Wi-Fi. Finally we have described the hardware based discovery approaches, such as low power wake-up circuits, passive and active RFID, accelerometers and passive infrared sensors. We described their advantages and limitations with respect to our scenario.

Chapter 3

Micro-level scheduling

3.1 Overview

Energy saving in wireless sensor networks is an issue of paramount importance. This is because in most scenarios the location of the sensors forbids both the use of constant power sources or the frequent replacement of the batteries. Energy scavenging solutions are also sometimes not sufficient, due, for example, to limited solar exposure of a solar panel. Most sensor networks therefore use some *duty cycling* to control the awake time of the sensors (or of their radio interface, which, often, is the most important source of power consumption).

Determining the perfect duty cycling is simpler if the sensors are fixed, and have a synchronized clock. A number of approaches exist for fixed wireless sensor networks, for instance S-MAC [Ye et al., 2004], which schedules all nodes in the network to wake up, listen and then sleep at the same time. D-MAC [Lu et al., 2004] and FPS [Hohlt et al., 2004] schedule the node wakeup times along a dissemination tree.

However if clocks cannot be synchronized (because clocks cannot be installed as there is a need for miniaturization or because non drifting clocks are expensive and cannot be afforded), other solutions need to be devised. Incidentally, these solutions often perform better than synchronous approaches. Examples of works, which have solved some of these issues include WiseMAC [El-Hoiydi and Decotignie, 2004], BMAC [Polastre et al., 2004] and, recently, X-MAC [Buettner et al., 2006]. In these solutions the nodes wake-up asynchronously and use carrier sensing to detect incoming transmission. Asynchronous duty cycling is very robust: it does not require fixed topology or very precise clocks and the long preamble is guaranteed to wake up any neighbouring node. The main drawbacks of asynchronous approaches have to do with the high discovery cost, because they require a long continuous transmission, known as preamble or packet train.

Fixed sensor network approaches [Ye et al., 2006] [Hohlt et al., 2004] are not directly applicable to mobile sensor networks because the connections between the nodes are often intermittent and the exact arrival time of the mobile node is uncertain. Therefore, the sensors have to be awake for long periods of time, wasting a lot of energy. Moreover, sparsely connected mobile wireless sensor networks have very different requirements from static networks. First of all, the nodes in sparsely connected networks spend considerable amounts of time *isolated* from the other nodes. To save power, mobile nodes need to be able to discover each other, while sleeping most of the time. Secondly, we expect a mobile sparsely connected network to have a very different traffic pattern. We expect the nodes to beacon frequently to discover other nodes and advertise own presence. A discovery would be followed by nodes transmitting their buffer contents, which involves a bulk data transfer. In sparsely connected networks there is often no direct path between source and destination and the nodes need to rely on gossiping [Haas et al., 2006], epidemic spreading [Vahdat and Becker, 2000] and flooding [Sasson et al., 2003] to deliver the data. This replication or synchronization traffic is likely to involve bidirectional interactive traffic, as nodes need to query each other about what data each nodes possesses before they replicate or exchange the data. Thirdly, the optimal operating point of a protocol must vary depending on the conditions. In other words, the protocol has to be *adaptive* to the changing traffic patterns. The amount of traffic generated by nodes can vary greatly depending on the environmental conditions. For example, in an application, where mobile nodes are used to track encounters between animals, the amount of data would vary greatly depending on time of day, season and other factors. The daily and seasonal variations affect the frequency and duration of encounters between the animals, which affects the amount of generated data records.

Problem formulation

The sparse mobile sensor network presents novel challenges for the design of energy efficient MAC protocols. First, global time synchronization is often not possible because existing time synchronization protocols are limited to connected subset of the network only $Q \in G$. The use of GPS on every node for global time synchronization is currently not practical due to cost concerns, and limited coverage in the scenario under consideration (deep foliage). Second, schedule synchronization between the nodes can potentially reduce power consumption but is hindered by limited precision of local clocks: diverging clocks on two nodes invalidate the common synchronized schedule. Third, the protocol needs to address the problem of timely discovery of neighbouring nodes by periodic beaconing.

We assume that the total energy consumption E_{total} is proportional to the active time of the radio. The goal is therefore to reduce the total time the radio is turned on either transmitting or receiving or idly listening:

$$E_{total} = E_{rx} + E_{tx} + E_{idle} \tag{3.1}$$

We assume that the network is sparse, i.e. consists of several disconnected subsets $Q_i \in G$. The set of nodes in each connected subset may change due to node mobility. The nodes need to perform periodic discovery requests to discover changes in topology. The discovery request requires the node to be awake for long period of time either listening or transmitting and is therefore considered an expensive process.

The goal of this chapter is to propose an asynchronous low power MAC protocol for wireless sensor network. The protocol solves the aforementioned challenges by using a number of novel mechanisms. The first mechanism is a pair-wise clock drift estimation for drift compensation between the nodes, enabling the nodes to synchronize their schedules for longer periods of time. Second, schedule exchange extension, allowing sparsely connected nodes to synchronize their schedules. A clock precision metric allows the nodes to estimate the schedule precision as it is passed in an opportunistic way among the nodes. Third, the protocol would support low power operation for pairwise traffic between any connected nodes.

MAC*ron*

In this chapter we propose MAC*ron*, an adaptive asynchronous duty cycling protocol. The protocol allows the nodes to duty cycle independently of each other, but lets the nodes learn the schedules of other nodes and exchange this information in a distributed way. This allows the nodes to send data packets precisely into the recepient's timeslots, without the need for either waking up the recepient with an expensive preamble or requiring all the nodes to synchronize to a common schedule.

The protocol addresses the problem of clock drifts, present in low-cost sensor nodes and proposes a lightweight synchronization mechanism, which estimates the relative clock drift between the nodes in the communication range. As opposed to time synchronization achieved by the use of dedicated time synchronzation protocols, our mechanism does not require periodic synchronization, nor extra communication traffic to keep track of neighbours schedules. The mechanism inserts timestamp information into each data packet, which is extracted by the recepient and used to calculate the drift.

The protocol also addresses the problem of advertising the schedules and the node presence to the rest of the network. Advertising the schedules requires sending a wake up signal followed by a schedule information. Any node in communication range can then respond within a certain time confirming its presence and specifying its own schedule in an acknowledgement.

This advertisement procedure is expensive and has to be performed periodically by each node. The protocol addresses this problem by proposing a schedule exchange extension. An extension allows the nodes to exchange known schedules between each other, so that it is enough for only one node in the neighbourhood to advertise schedules. An extension should reduce consumption, as well as contention in dense environments. In heterogeneous environments, it can be a powerful node, or a sink, responsible only for waking up and helping nodes to discover each other. The protocol learns the clock drifts between pairs of individual nodes and propagates the clock information combining the information from the most stable nodes. We introduce a new cost metric reflecting the clock stability and demonstrate how the problem can be solved using any standard routing protocol.

We evaluated the protocol in real setting and compared its performance to recent MAC protocols for typical data collection or data dissemination scenarios. The protocol has been implemented for Chipcon[1] CC2420 radio on the TMote Sky platform [MoteIV, 2006] using TinyOS-2.x operating system and tested on a small scale testbed.

The rest of the chapter is structured as follows. In Section 3.2 we describe the protocol in details. Section 3.3 contains the description of Schedule Exchange Extension in dense scenarios. Section 3.4 reports on the details of our MAC*ron* implementation. Section 3.5 discusses related work and Section 3.6 concludes the chapter. The evaluation of the protocol is presented in Chapter 5.

3.2 MAC*ron* Design

We first argue about the advantages of asynchronous duty cycling and motivate our design choice for the MAC*ron* protocol. We then argue that even though our protocol does not rely upon precise clocks, it can still exploit this information, when available, to improve its performance.

Robustness, simplicity and fault tolerance are the major properties of asynchronous duty cycling. In asynchronous operation, nodes schedule their wake up time independently of each other, initiating a communication session, a node uses a continuous modulated tone, which is guaranteed to wake up any nodes in communicationn range. This operation is completely asynchronous, and does not rely upon any complex global time synchronization protocols, expensive oscillators or GPS.

On the other hand, global time synchronization is crucial for power conservation and its importance has been discussed extensively in sensor network literature. At MAC layer, precise

[1]Now Texas Instruments.

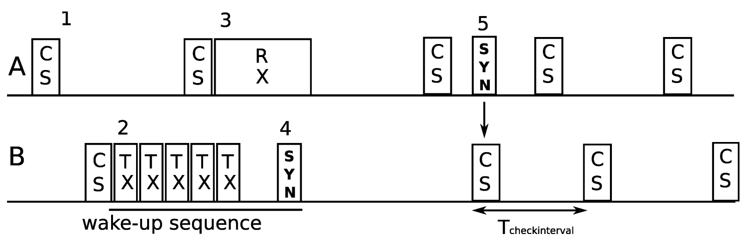


Figure 3.1: Wake-up scheduling: 1 - each node wakes up at regular intervals. 2 - selected nodes periodically wake up their neighbours with extended preamble. 3 - node A detects transmission and wakes up. 4 - node B broadcasts a SYNC message containing schedule information. 5 - node A sends an ACK at B's timeslot.

time synchronization allows scheduling the communication sessions between the nodes, thus reducing the idle listening time and obviating the need for energy intensive wake-up signal, because the nodes are programmed to wake up at fixed known timeslots.

While crucial for energy efficient operation, time synchronization in sparse mobile networks can be a difficult task. Global time synchronization requires either each node to be equipped with expensive GPS or use of time synchronization protocols. GPS has limited coverage under certain conditions, such as deep foliage. The use of time synchronization protocols is not always possible in mobile wireless sensor networks, where, due to dynamic network partitioning, a group of nodes might end up without a reference clock. Time synchronization within the partition would require selecting a cluster leader, a local reference clock, which would be regarded as 'true' clock in the partition. These cluster leaders would have to be re-elected every time the partition splits or merges or the partitions would have to maintain several schedules. This would either complicate the protocol or lead to excessive energy consumption.

Therefore our design choice is to use asynchronous duty cycling as a basis and use precise clock information, whenever such information is available. The protocol includes a simple mechanism to exploit and propagate the clock information to all the nodes in the network.

3.2.1 Protocol at a Glance

We now present the basic operation of the protocol. The mobile sensors initially have independent schedules of wake up time and are not aware of each other's presence. The main steps of the protocol are:

1. Each node wakes up at regular intervals $T_{checkInterval}$ to perform carrier sensing. Since each node follows its own schedule, there is a fixed phase offset between neighbouring nodes.

2. Each node periodically advertises its own schedule by waking up neighbouring nodes with an extended preamble followed by a SYNC packet containing local schedule offset and a local timestamp (Figure 3.1). The basic assumption is that the extended preamble is longer than $T_{checkInterval}$.

3. The neighbouring nodes detect a preamble during a regular carrier sensing, wake up and receive a SYNC packet.

4. Upon receiving a SYNC packet, the nodes calculate the new node's schedule by subtracting the schedule offset from the timestamp from the SYNC packet.

5. A neighbouring node can now send traffic packets to the node by waking up on purpose at the right time (as indicated by the schedule) when in need to communicate with that specific node. Note that the original schedule of each node is only altered when in need to speak to a specific other node and returns to the original after that communication has occurred.

6. Upon receiving a SYNC packet, a node responds with a similar SYNC packet with a random delay on the new node's schedule.

The procedure ensures all nodes know each other's schedule and can send packets at strictly each other's timeslots. The approach does not require global time synchronization. It also reduces interference between the neighbouring nodes because each pair of nodes in the same broadcast domain are likely to use different (or unique) timeslots.

Because of the clock drift, and limited clock precision, the relative offset between the nodes will change over time. Thus, to wake up the neighbouring node, the length of the preamble has to account for the clock error accumulated since the last received packet from recepient node. The low cost nodes have a limited precision clocks, so the clock error can accumulate quickly over time. For example, the typical clock precision of the sensor nodes is 50ppm ($\pm 25ppm$), which translates to the clocks diverging by 4.3s over a day. This puts a lower boundary on the duty cycling as the nodes have to use either longer preambles to account for clock divergence or receivers have to perform more frequent carrier sensing.

In high traffic scenarios, schedule information can be sent relatively regularly (either in separate packets or piggybacked as a separate field on data packets) so that nodes can correct the offset and continue to be able to communicate. However if the communication is less frequent, the nodes would have to use longer preambles to compensate an accumulated clock error, which results in higher transmission costs. In our approach, the protocol passively monitors all

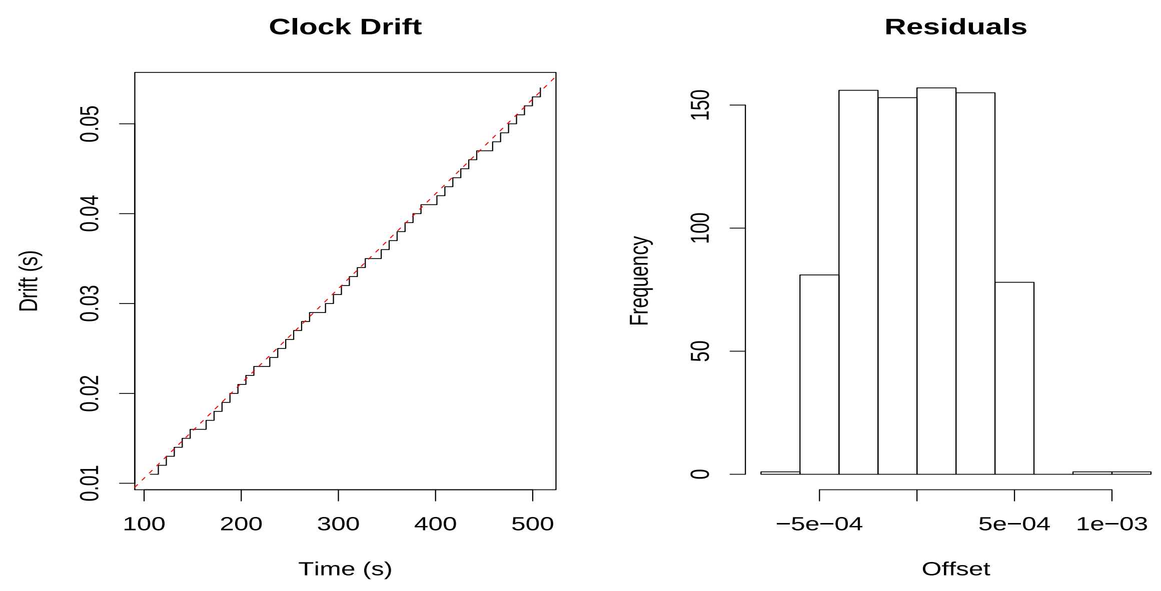


Figure 3.2: Clock drift graph.

the traffic between the nodes and estimates the relative drift between the nodes to correct the schedules of the nodes.

We now describe the pairwise passive drift estimation, which helps to compensate for errors in calculating the neighbour's schedules over single hop.

3.2.2 Pairwise Drift Estimation

A number of measurements we conducted both indoors and outdoors (Figure 3.2) shows the linearity of the clock drift. The drifts and the linearity are also consistent with a number of studies on time synchronization protocols [Ganeriwal et al., 2003, Maroti et al., 2004]. Thus, we model the clock as an oscillator of a fixed but unknown frequency dependent on a variety of factors such as temperature and humidity. The relative drift between the clocks is modeled as a linear process:

$$\delta(t) = \theta t + \epsilon(t) \tag{3.2}$$

Where $\delta(t)$ is an absolute drift between the two clocks at time t, θ - is a relative clock drift that needs to be estimated, $\epsilon(t)$ - is a Gaussian noise.

Estimation of clock drift θ can be done using standard linear regression statistical methods, a method also used by some dedicated time synchronization protocols [Ganeriwal et al., 2003, Maroti et al., 2004]. Linear regression tries to estimate a general trend of all data points by fitting a line (Figure 3.3), in a way that 'best' explains the general trend of all points. The best fit is found by minimizing a certain criteria, such as a total sum of geometrical distances between the line and the point.

The popular minimum least squares estimation (MLSE) method used in previous works on time synchronization method is memory expensive as it requires storage of all data points. While providing a good estimation, the high memory usage is undesirable for resource constrained sensor nodes. Such method is also computationally expensive as adding a new data point requires recalculation of a solution from scratch.

MAC*ron* uses a very lightweight and computationally efficient recursive least squares linear regression method, used extensively in signal processing, communications and control applications. We describe it in the next subsection.

Recursive Least Squares

Recursive least squares (RLS) is a computationally efficient algoritm to find linear regression coefficients while minimizing the weighted squared error:

$$J = \sum_{t=0}^{T} (\lambda^{t-k}(y(t) - \theta(t)\phi(t))) \tag{3.3}$$

Where J is a weighted square error. Expression 3.3 shows a weighted squared error between the actual ($y(t)$) and the predicted $\theta(t)\phi(t)$ values of time. Where $y(t)$ is a relative offset between the clocks, $\theta(t)$ is a drift, the value of which needs to be estimated, and the $\phi(t)$ is the amount of time elapsed from the last clock correction. The goal is to find an optimal value of $\theta(t)$, which would best estimate the actual drift between the nodes.

RLS is particularly suitable for resource-constrained devices and real-time applications: given a new sample $y(t)$, a new estimate is calculated in a fixed number of steps independent of t, making it very efficient. The recursive form of the solution is given by the following equations:

$$\theta(t) = \theta(t-1) + K(t)\epsilon(t) \tag{3.4}$$

$$\epsilon(t) = y(t) - \phi(t)^T\theta(t-1) \tag{3.5}$$

$$K(t) = P(t-1)\phi(t)[\lambda I + \phi^T P(t-1)\phi(t)]^{-1} \tag{3.6}$$

$$P(t) = [I - K(t)\phi^T]P(t-1)/\lambda \tag{3.7}$$

The current estimate $\theta(t)$ is calculated (Eq 3.4) by adding correction to previous estimate $\theta(t-1)$. The amount of correction is proportional to the error $\epsilon(t)$, which is the difference (Eq

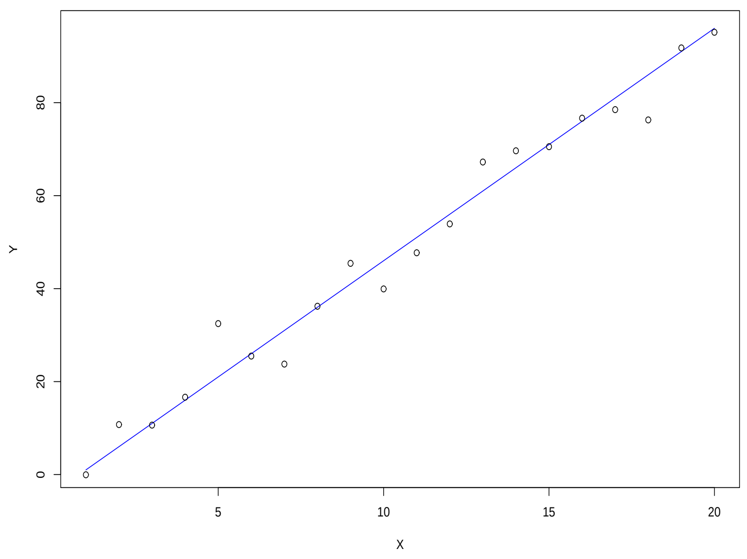


Figure 3.3: Linear regression estimation.

3.5) between the model output $\phi(t)^T\theta(t-1)$ and the actual input $y(t)$. The parameter λ is a 'forgetting factor' giving more emphasis to new data points. The value of the forgetting factor determines the weight of past samples and its selection is a trade off between noise immunity and ability to adapt to changes. The value of the forgetting factor was set empirically to 0.95. The algorithm is easily implemented even on severely resource constrained sensor platform and calculating a recursive solution involves just N subtractions and N divisions. The algorithm was implemented using floating point arithmetics external library. To avoid dealing with very small numbers, we calculate $1/\theta$ instead, which can be expressed in positive integers. Thus the algorithm is estimating the amount of time, in seconds, it takes to accumulate a 1ms drift. It is easy to convert an algorithm to fixed point arithmetics for further performance improvements.

The algorithm implementation takes into account the time measurement quantization, as the sensor nodes measure in milliseconds, truncating the real time to the nearest millisecond. The quantization error in calculating the offset based on two points is:

$$d = (\omega_2 \pm e_2 - \omega_1 \pm e_1) = (\omega_2 - \omega_1 \pm e_2 \pm e_1) \tag{3.8}$$

Thus, the result from calculating few closely spaced points result in large degree of error. The algorithm uses all timestamps for drift estimation, but starts correcting the schedule only once the timestamps are separated by at least 120 seconds, which limits the quantization error down to 0.5ms per minute.

The node can use the information about the drift to estimate the schedules of neighbouring nodes, which we describe in the next subsection.

3.2.3 Schedule Correction

We will now describe how the nodes use pairwise drift information to precisely estimate the schedules of neighbouring nodes. Normally, all the nodes in the network have strictly periodic schedules and wake every $T_{checkInterval}$ seconds. Since the nodes measure the time using the local clocks, the actual schedule interval will be longer or shorter depending on the drift. If the neighbour's clocks are faster, the actual schedule interval becomes shorter, which means the node has to correct the neighbour's schedule offset by subtracting a correction. Similarly, if the neighbour's clocks are slower, the correction has to be added to the schedule offset. This is summarized by the following expression:

$$S = (S_0 + S_{comp}) \bmod T_{checkInterval} \tag{3.9}$$

$$S_{comp} = (\phi(t) - \phi(t_0))\theta(t) \tag{3.10}$$

Where S_0 - the previous known schedule value, S_{comp} - an additional offset caused by the clocks, $(\phi(t) - \phi(t_0))$ - the time since the last packet (schedule) was received.

The drift is estimated based on the timestamps and schedule information from the incoming packets. For each received packet, the node feeds an $[node\_id, offset, time]$ tuple to the drift estimation module. Thus, the drift is only estimated if there is a communication traffic between the nodes. This is in contrast to existing time synchronization methods, which usually run a three packet handshake session complicate the protocol by introducing an extra state and traffic. The drift estimate will be progressively improved as there is more communication between the nodes. The tighter drift estimate will, in turn, lead to shorter preambles and less transmitted energy consumption.

In the next subsection we will describe how drift estimation affects the wake up preamble length.

Preamble Length

Without the drift estimation the wake up preamble has to be long enough to account for relative drift between the nodes. The preamble length is calculated according to the following expression:

$$L_{preamble} = (t_{now} - t_{lastHeard}) \times 2\theta(t) \tag{3.11}$$

Where $t_{now} - t_{lastHeard}$, is the duration since the node last heard a schedule update from the neighbour i.e. the last time a packet was received from a neighbour. $\theta(t)$ is a relative drift between the two nodes.

As soon as the node has information about the clock drift, the preamble length can be much shorter and has to be long enough to compensate for residual errors in drift estimation. The actual preamble length includes a confidence interval, depending on the desirable reliability of wake-up. The confidence interval was set to $\delta_{95\%}$, but can be increased depending on the application requirements.

$$L_{preamble} = (t_{now} - t_{lastHeard}) \times 2(\theta(t) \pm \delta_{95\%}) \tag{3.12}$$

Thus, the preamble length can be much shorter and result in significant savings in transmission power for low data rate communications.

3.3 Schedule Exchange Extension

A Schedule Exchange Extension addresses the problem of schedule advertisements in dense environments. Advertising a schedule is expensive as it requires the nodes to send an extended preamble periodically to wake up and advertise its presence to potentially new neighbour nodes. It is clear that in dense environments, there is no need for all nodes to advertise their schedules. It is sufficient for $1/n$ nodes to perform this process and then exchange this schedule information among the neighbouring nodes. This reduces the transmission costs by n times and significantly reducing interference in the network. Another advantage is significant reduction in reception costs, as each preamble is received by every node in the broadcast domain.

We illustrate further benefits of schedule exchange on a wildlife tracking scenario presented on Figure 3.4. In this scenario, the mobile tags attached to badgers are powered on a small battery, which needs to last for as long as possible. More powerful nodes are installed at setts and places, visited periodically by badgers. For this scenario, it is sensible to shift all communication and node discovery load onto more powerful nodes, where battery can be replaced or recharged more easily. The default schedule advertisement (and node discovery procedure) is therefore as follows. A powerful node C sends periodic waking up signal followed by its schedule information. The mobile tags A and B respond and can then communicate with the base station.

More specifically, Node C advertises its schedule using an extended preamble to both A and B. This results in communication links AC and BC. Without schedule exchange, nodes A and B can communicate through node C only. However, nodes A and B can calculate each

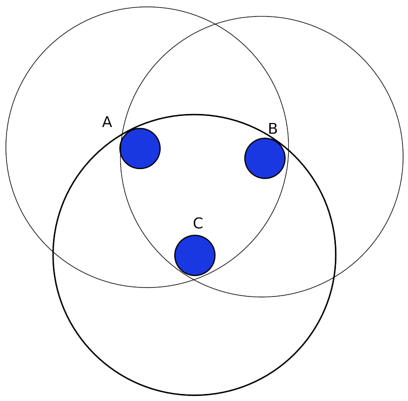


Figure 3.4: Schedule exchange diagram. Node C advertises its schedule using an extended preamble to both A and B. Nodes A and B can learn each other's schedules from C.

other's schedule by knowing relative clock drifts from C to A and B. For example, $\theta_{BA} = \theta_{CA} - \theta_{CB}$. The problem is that the mobile tags still cannot communicate with each other unless they start sending periodic wake up signal themselves, which would seriosly limit the lifetime.

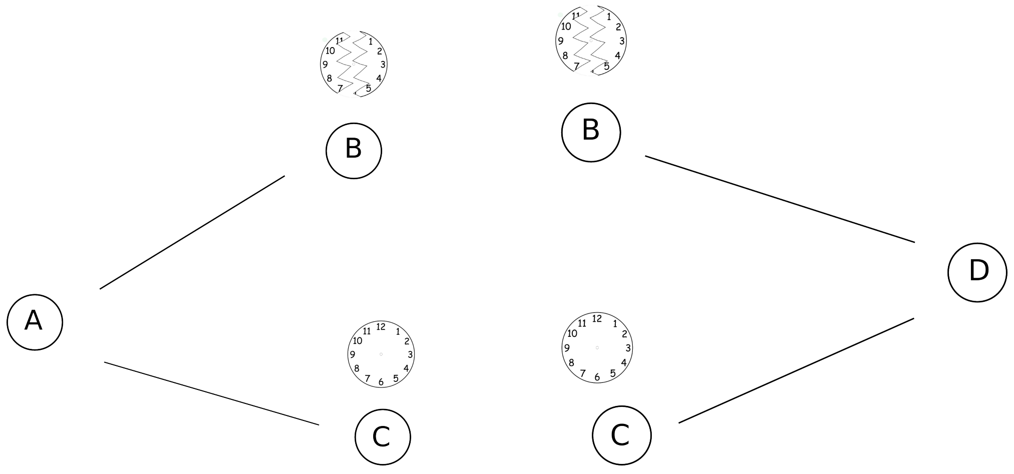
The Schedule Exchange Extension lets the nodes communicate directly with each other without using wake up signals. The communication relies on more powerful nodes, which serves the purpose of node discovery and can also serve as a base station. To summarize, the nodes can exchange schedule information to reduce the transmission, reception costs as well as interference in the network. The schedule information, clock drift and offsets of known nodes is contained in the SYNC message. The nodes calculate the schedules and clock drift of other nodes in the network. This enables the nodes to communicate directly instead of relaying data through other nodes.

We now describe the problem arising in drift estimation during schedule exchange and propose a novel distributed schedule selection protocol (Subsection 3.3.1).

3.3.1 Schedules Selection Problem

The problem of schedule selection is illustrated on the Figure 3.5. At point t_1, node A communicates with nodes B and C, which learn A's schedule and estimate A's drift. Some time later (t_2), nodes B and C move over to D and can advertise A's schedules and clock drifts to D.

As nodes B and C do not transfer the schedules to D immediately, but after a certain time delay, this schedule information gets affected by the local clock drift of both nodes. The degree to which the schedules are affected depends on the duration of this delay, as well as the quality of node B's and C's clocks. The problem is whether the node D should accept A's schedules from B or C or from neither of the nodes.



(a) Nodes B and C learn A's schedule at t_1.

(b) Nodes B and C pass A's schedule to D at t_2.

Figure 3.5: Schedule selection problem.

3.3.2 Clock Quality Metric

To solve the schedule selection problem, we introduce a *relative clock quality metric*, which reflects the stability of clocks along the communication links between the nodes in the network. The clock quality C_{ij} of the communication path between nodes i and j is:

$$C_{ij} = St_j \times \sigma_{ij} \times T_{age,ij} \tag{3.13}$$

Where St_j is a clock precision in ppm[2] based on clock specifications. It shows, for example, that the schedules propagated along the nodes with perfect precision clocks ($St = 0$) will be always preferred to schedules propagated along other paths. σ_{ij} is a variance in relative clock measurement between the nodes along the communication link, it is needed to give preference to paths where nodes have the lowest variance in clock drift estimation. The higher values of variance σ might indicate clock instability due to environmental changes or measurement noise. Currently, it only reflects the clock instability over a period of time when the nodes are communicating. T_{age} reflects the age of the link and will prefer fresher paths in the network.

The clock quality metric allows to define a simple criteria to select one schedule over the other. We will now present a distributed schedule selection algorithm based on a clock quality metric. We first explain an algorithm and then discuss its relation to distributed Bellman-Ford algorithm for shortest path calculation in connected networks.

[2]parts per million.

ID	Schedule ID	Offset	Drift	Cost/Quality
1	Node A	1000ms	40ppm	0
2	Node B	200ms	30ppm	10
3	Node C	200ms	0ppm	10

Table 3.1: An example of a scheduling table.

3.3.3 Distributed Schedule Selection Algorithm

As mentioned earlier, each node advertises the list of known schedules in a SYNC message. The SYNC message is advertised whenever the nodes first discover each other.

Upon receiving a SYNC message, a node adds a local clock quality metric to each of the schedules in the list to reflect the quality of the current link. The node then starts merging the schedules in the following way. Any schedules for previously unknown nodes are simply added to the list of known schedules. For destinations, which already exist in the list of known schedules, a node compares the clock quality metrics of both schedules and keeps the schedule with the lowest metric. This ensures the propagation of schedules along the paths with the lowest clock metric, or in other words, along the nodes with the most stable clocks.

An example of a scheduling table is shown in Table 3.1. Each schedule entry contains information about neighbour's relative clock drift and a clock precision. The table shows, for example, that the information about node A's schedule is more precise despite it having a higher clock drift than other nodes. The lower clock metric allows to estimate A's schedule with higher precision, which translates into lower transmission and reception costs for waking up the node. The nodes repeat the schedule update procedure every time they meet a new node. The procedure guarantees that the scheduling information of all nodes will eventually converge to the best values.

3.4 Implementation

In this section we describe the implementation details of MAC*ron* protocol.

MAC*ron* has been implemented for Tmote Sky sensor nodes [MoteIV, 2006] in TinyOS-2.x. TinyOS is an event-based operating system designed for networked embedded applications. We chose TinyOS for a number of reasons. Firstly, it is an open source operating system and is available for free. Secondly, it allows building very light-weight applications with minimal overhead. Third, it is well documented and contains driver implementations for many types of devices. TinyOS has become a de facto standard in sensor network community and has a very large user base. The protocol was implemented in NesC, a C dialect with a special support for components. The entire protocol was implemented as a component, which in turn integrates several smaller components that implement scheduling, controlling and neighbour management functions.

The sensor node contains a modern low-cost 802.15.4 compliant Chipcon CC2420 radio [TexasInstruments, 2008] designed for low-power wireless applications. The radio provides raw data rates of 250kbs and has very fast start-up time. The chip consumes 18.8mA in receiving mode and 17.4mA in transmitting mode. The radio consumes less than 1μA in power down mode and can remain in that mode until a transmission or scheduled reception is requested.

The implementation contains 25kB of ROM and 2.5kB of RAM. It consist of *scheduling*, *control*, *drift estimation* and *low power listening* (LPL) modules. The *scheduling module* manages asynchronous and synchronous schedules a node maintains. The module provides primitives to add, remove, lookup and modify schedules. A node consults the schedule module about the next wake-up time each time it goes to sleep. Each schedule consist of a tuple containing a session start time and duration. The *scheduling module* also maintains a global node state (such as sampling, sending and idle) to coordinate between various modules. The *control module* performs neighbour management, adaptation, synchronization and LPL backoff functionality.

A difficult problem turned out to be occasional freezing, due to race conditions in the radio driver. Such bugs would only appear only after several hours of operation, making them very difficult to localize and debug. The use of simulators allowed to debug the protocol logic, but did not provide support for the actual hardware, for example CC2420 chip is only supported at the packet level. Also, the simulators do not take into account timing issues, which are often the cause of hardware lockups and timeouts. Debugging was done by making the problem happen earlier in the experiment and localizing the problem by inserting debug statements over serial port.

Finally, collisions have presented greater problem than expected in scenarios with more than four senders. The long check intervals were important to achieve very low duty cycle, but on the other hand, tended to increase collisions, as senders tended to compete for fewer timeslots. Randomizing the initial node schedules and beaconing intervals was very important to avoid and minimize the impact of collisions.

The LPL functionality is achieved by repetitive transmissions of the packets for the entire length of the sleep interval T_{sleep}. To minimize the gaps between the packets, which could be misintepreted by receiver as clear channel the sender disables ACKs, initial and congestion

Table 3.2: CC2420 power consumption

State	Power Consumption
transmit, P = 0dBm (max)	17.4mA
transmit, P = -25dBm (min)	8.5mA
receive	19.7mA
idle mode	426μA
power down	20μA

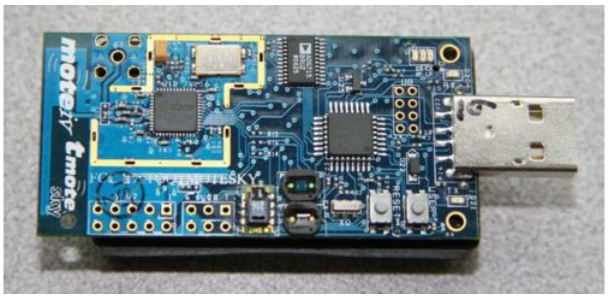


Figure 3.6: TMote Sky sensor node.

CSMA backoffs and sends the packets back-to-back as quickly as possible by issuing STXON command as soon as the packet is transmitted. The STXON command causes the radio to retransmit the last packet without loading a new packet into TXFIFO buffer. Upon detecting a preamble a receiving node starts sampling more frequently until the preamble is finished and then waits for synchronizing message.

Precise time synchronization between sender and receiver is done using Start Frame Delimeter (SFD) pin information, which indicates that the physical header of the 802.15.4 packet has been completely transmitted. When sending a synchronization packet a sender adds a pending schedule to its table of schedules. As soon as SFD signal is fired, the pending schedule is adjusted to start from that moment. Upon receiving a synchronization packet, a receiver immediately adds this new schedule to its table of schedules. A receiver wakes up for each synchronous slot, and can skip regular carrier checks if the interval between succesive wakeups is less than the current check interval. The implementation does not use any features specific to CC2420 radio, so it could be easily ported to other hardware and software platforms.

3.4.1 Neighbour Table Format

The implementation provides neighbour management functionality. It allows an application to define proximity, handles beaconing, notifies an application on node discovery and disconnections. This is convenient for mobile application development because it relieves the developer from low-level tasks.

The neighbourhood table (Table 3.3) contains the soft state information about the neighbours. After a certain timeout defined by the application, the neighbour is marked as inactive but not removed from the table. The timeout depends on the bit error rate of the channel and reflects how many beacon can be missed before the neighbour is considered disconnected. Encounter tracking experiments [Scott et al., 2006] discovered that a very large number of encounters were only separated by 2 beacon intervals and suggested that the contact period stops after 2 successively missing beacons. The field *thetaEstimate* field contains a current estimate of the relative clock drift of a node. *lastHeard* value keeps record of when a node last heard from this neighbour. The field is used to calculate the schedule offset correction and preamble length. The fields *LQI* and *RSSI* fields contain current estimates of link quality and signal strength respectively. Each neighbour also contains a number of variables for recursive least square linear regression.

The neighbourhood table is extendable and can potentially contain a number of other attributes including frequency of encounters and other historical information to facilitate the operation of mobile applications and routing protocols.

Table 3.3: Neighbour Table Format

Field	**Type**	**Description**
node ID	uint16_t	node ID
lastHeard	uint16_t	time last heard from neighbour
thetaEstimate	uint32_t	current drift estimate
LQI	uint16_t	link quality, lqi
RSSI	uint16_t	received signal strength, rssi

Table 3.4: Duty Cycle API

Function call	**Type**
void setInterval(uint16_t id)	set channel sampling interval *id*
void setBeaconInterval(uint16_t id)	set beacon interval *id*
uint16_t getInterval()	get channel sampling interval *id*

Table 3.5: Proximity API

Function call	**Type**
uint8_t present(uint16_t id)	check if neighbourhood table contains an entry for the node *id*
bool synced(uint16_t id)	check if the neighbour is synced
void neighbour_update(uint16_t id)	check if the neighbour is synced
void disconnect(uint16_t id)	callback when a node disconnects
void maxGap()	set maximum gap allowed in the encounter
uint16_t encounter(uint16_t ts1, uint16_t ts2)	callback when a disconnects permanently

3.4.2 Programming API

3.4.2.1 Duty Cycling

The implementation provides a set of APIs that allows applications to control the parameters of duty cycling process (Table 3.4). The application can change channel check interval and beacon interval. *setInterval(uint16_t id)* sets the channel sampling interval. The sampling interval can range from several hundred milliseconds to several seconds, depending on the amount of traffic and desired duty cycle. For example, for ultra-low duty cycle level of 0.3% , the protocol is configured with a sampling interval of 3 seconds.

setBeaconInterval(uint16_t id) sets the beaconing interval. This parameter depends on the level of network mobility. Low mobile applications require infrequent beaconing, whereas high mobility applications require frequent beaconing to keep track of dynamic changes in topology. The beacon interval can be adaptive depending on the mobility patterns and is controlled by the *macro* layer, which we will introduce in the next chapter.

The process of duty cycling, a periodic wake-ups and sleeps, is completely transparent to the application. The application layer, however, has a complete control over the parameters of the duty cycling process.

3.4.2.2 Proximity API

The implementation provides a compact API for neighbour proximity tracking (Table 3.5), a useful feature for mobile applications, as it helps keep track of the neighbour presence. The proximity API provides a callback function to notify an application that the node has disconnected, *disconnect(uint16_t id)*. An application can set a timeout after which a node is considered disconnected, *maxGap()*, the timeout depends on the bit error rate of the channel and

Table 3.6: Logging API

Function call	Type
void sum32(uint32_t x)	Add *x* to the sum, used for logging uptime
void count(uint16_t id)	Count number of packets from *id*
void flush()	Flush serial buffers

reflects how many beacon can be missed before the neighbour is considered disconnected. Finally, each encounter is logged using *encounter(uint16_t ts1, uint16_t ts2)* callback function, where *ts1* and *ts2* are the times when the encounter was detected and stopped respectively.

3.4.2.3 Logging API

The implementation has a custom logging module to collect statistics such as the number of received packets, total time spent in sleeping and active modes and debugging information. The results are reported by each node over USB interface at the end of each experiment. The API contains *sum32(uint32_t x)* , *count(uint32_t x)* and *flush()* function calls as shown in Table 3.6.

We will now discuss how MAC*ron* relates to existing work.

3.5 Related Work

The most relevant related work to MAC*ron* is on energy efficient MAC protocols in wireless sensor networks. A number of approaches have been proposed including SMAC [Ye et al., 2004], DMAC [Lu et al., 2004], BMAC [Polastre et al., 2004], SCP-MAC [Ye et al., 2006] and X-MAC [Buettner et al., 2006]. The approaches can be split into three categories: synchronous, asynchronous and recently a hybrid approach. Most approaches rely on static topology and consider node discovery mostly at deployment or configuration stages. In the next subsections we briefly describe these approaches, a more detailed comparison with some of these approaches, such as BMAC and X-MAC is presented in the evaluation section.

3.5.1 Synchronous MAC Protocols

Determining the perfect duty cycling is simpler if the sensors are fixed, and have a synchronized clock. Synchronous approaches do not totally eliminate the idle listening time, but help to reduce it. Most synchronous approaches are based on establishing communication schedules, so that nodes communicate on known timeslots.

A number of approaches exist for fixed wireless sensor networks, for instance S-MAC [Ye et al., 2004], which schedules all nodes in the network to wake up, listen and then sleep at the same time. D-MAC [Lu et al., 2004] and FPS [Hohlt et al., 2004] schedule the node wakeup times along a dissemination tree. The tree is constructed during configuration phase and then nodes are assumed to be static thereafter.

IEEE 802.11 MAC provides built-in support for power management. The standard provides two power management states: active mode (AM) and power save mode (PS). In active state, the nodes are always awake and ready to receive data. In power save mode the base station buffers the packets for the nodes in power save state. The nodes sleep most of the time and periodically wake up and check for incoming packets. Such an approach is suitable for infrastructure based mode with an access point.

The protocols designed for fixed networks do not consider interactions between mobile nodes and mobile groups of nodes. A global network synchronization, as it is done in SMAC or SCP-MAC for example, might lead to a situation where two or more groups of synchronized nodes will never see each other because of constant phase shift between individual groups of sensors. This is not a problem in static scenarios, where the nodes need to discover each other only once, during configuration phase or failure, however, in a mobile scenario, the lack of efficient discovery mechanism would lead to a network partitioning.

Complexity is another drawback of synchronous approaches. The network needs to be periodically resynchronized using built-in synchronization support or time synchronization protocols, or high precision hardware, such as GPS or atomic clocks. Finally, the synchronous approaches are inherently less fault-tolerant, as nodes, which loose synchronization become isolated from the network.

3.5.2 Asynchronous MAC Protocols

Asynchronous approaches [El-Hoiydi and Decotignie, 2004], [Polastre et al., 2004] have been developed and tested in static wireless sensor networks. These approaches are robust: they do not require fixed topology or precise time synchronization and therefore more suitable for mobile scenarios. A number of optimizations proposed to asynchronous approaches targeted its performance in dense environments and high-throughput applications. X-MAC [Buettner et al., 2006] and WiseMAC [El-Hoiydi and Decotignie, 2004]. WiseMAC optimizes low power listening by making senders learn the wake-up schedules of their neighbours. The nodes thus can use very short preambles to selectively wake up a specific neighbour.

The main drawbacks of asynchronous approaches have to do with the high discovery cost. Discovering neighbouring nodes usually requires staying awake for a longer periods of time either continuously transmitting beacons or listening for beacons from potential neighbours.

There have been a number of recent attempts to apply asynchronous duty cycling to mobile wireless networks such as [McGlynn and Borbash, 2001], [Zheng et al., 2003], [Jiang et al., 2005], [Jun et al., 2006], [Borbash et al., 2007]. These approaches are based on generating a certain sequence of states between listening, transmitting and idle states. The sequences are designed to maximize the probability of overlapping between the node active states. [McGlynn and Borbash, 2001] looks at energy efficient node discovery after sensor nodes are deployed in the field. [Zheng et al., 2003], [Jiang et al., 2005] and more recently [Borbash et al., 2007] investigate an efficient way for mobile nodes to discover neighbours. These approaches assume that the minimum awake time of node is enough to receive an entire packet (beacon). The assumption is not correct for modern radios, where transmission detection through channel sampling can be done in much shorter time than receiving entire packet.

3.5.3 Hybrid MAC Protocols

SCP-MAC [Ye et al., 2006] uses a technique called scheduled channel polling. The technique allows to reduce the cost of wake-up preamble by synchronizing entire sensor network. The protocol achieves very low power consumption under certain conditions. Synchronizing entire network might be unnecessary or undesirable in certain situations. It requires either running a special synchronization procedure or constant periodic traffic between all nodes to piggyback schedule information. MAC*ron* synchronizes the nodes on demand, only when communication is needed. Thus it is conceptually simpler and easier to implement in practice. MAC*ron* is more suitable to heterogeneous and mobile environments, where not all the nodes are allowed to beacon because of energy constraints or where global time synchronization is not available.

X-MAC [Buettner et al., 2006] uses a number of techniques to minimize energy consumption due to long preambles. In X-MAC the nodes nodes use a sequence of packets sent back-to-back to wake up neighbours. The X-MAC proposes an optimization, where the nodes can stop this packet train early by sending an ack to the sender. The drawback of this scheme is that the sender has to listen in between the short packets composing a preamble. This results in longer clear channel assessment (CCA) checks for the receiver, which makes it less efficient. A second limitation is that a receiver would still need to receive on average half a preamble before the start of a data transmission.

3.5.4 Other Approaches

Other duty cycling approaches include STEM [Stemm and Katz, 1997], SPAN [Chen et al., 2001], PAMAS [Suresh Singh and Cauligi Raghavendra, 1999], GAF [Xu et al., 2001], RMAC [Du et al., 2007] and [Nath and Gibbons, 2007] were mainly developed for static networks and are briefly mentioned below. STEM [Stemm and Katz, 1997] is a MAC protocol, which uses a separate low power radio for signaling purposes. The nodes receives a request on a separate channel and wakes up the main radio to receive a packet. Thus, it does not require time synchronization and can wake the nodes on-demand. SPAN [Chen et al., 2001] is an approach, which identifies multiple network subsets, each subset providing entire network connectivity. PAMAS [Suresh Singh and Cauligi Raghavendra, 1999] is another MAC protocol, which tries to reduce power consumption by turning off radio when it overhears packets for other nodes. This avoids energy consumption related to processing of incoming packets. GAF [Xu et al., 2001] splits the network into square grid, where all nodes in a cell can communicate with each other. Since the communication within a cell can be provided by any node, the rest of the nodes can be put to sleep. Recent approaches include cross-layer integration of wake up scheduling with routing protocols such as [Nath and Gibbons, 2007], which integrates wake up scheduling with geographical routing, RMAC [Du et al., 2007], which uses cross-layer integration to reduce end-to-end latency, [Huang and Yang, 2007] propose a double sense multiple access protocol, which reduces contention and the network power consumption while increasing the system throughput. RI-MAC [Sun et al., 2008] proposes an asynchronous duty cycling protocol that uses receiver initiated data transmission to improve throughput and packet delivery ratio.

3.5.5 Summary of Existing Work

Fixed sensor network approaches are not directly applicable to mobile sensor networks because the connections between the nodes are often intermittent and the exact arrival time of the mobile node is uncertain. Therefore, the sensors have to be awake most of the time consuming a lot of energy. Moreover, sparsely connected mobile wireless sensor networks have very different requirements from static networks. First of all, the nodes in sparsely connected networks spend considerable amounts of time *isolated* from the other nodes. To save power, mobile nodes need to be able to discover each other, while sleeping most of the time.

Secondly, we expect a mobile sparsely connected network to have a very different traffic pattern from a static network. In sparsely connected networks there is often no direct path between source and destination and the nodes need to rely on gossiping [Haas et al., 2006], epidemic spreading [Vahdat and Becker, 2000] and flooding [Sasson et al., 2003] to deliver the data. This replication or synchronization traffic is likely to *involve bidirectional interactive traffic*, as nodes need to query each other about what each nodes has before they replicate or exchange the data.

3.6 Summary

We have presented MAC*ron*, a wake-up scheduling protocol for mobile wireless sensor networks.

The protocol is using novel schedule exchange algorithm and passive pairwise drift estimation to reduce power consumption. In the protocol, the nodes duty cycle asyncronously, and learn the schedules of other nodes. To save transmission energy, the nodes compensate for relative clock drift between the nodes using a lightweight clock-skew estimation algorithm. For the dense scenarios, the protocol uses a novel schedule exchange extension, which reduces the number of expensive broadcasts. A schedule extension works by distributing the schedules of known nodes in every beacon packet, advertising not only own but also the schedules of other nodes. The protocol introduces a novel *clock quality metric*, which minimizes the clock aberration while transferring timestamp information through nodes with drifting clocks. Finally, the protocol has been implemented for Tmote Sky sensor nodes on TinyOS operating system. We report the evaluation of the protocol in Chapter 5.

Chapter 4

Macro-level Scheduling

4.1 Overview

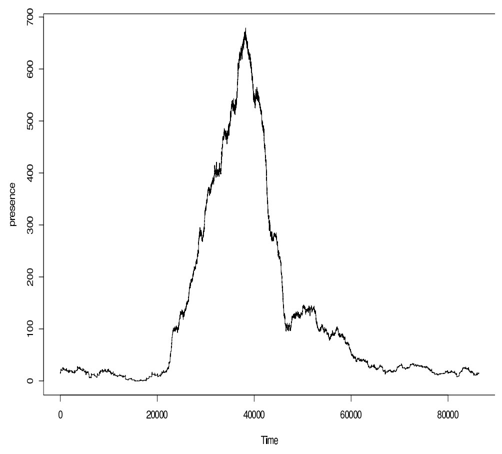
The micro-level approach is robust because it does not rely on any assumptions about mobility pattern of nodes: if collocation time is above a certain threshold, a micro-level scheduling guarantees detecting a node within a communication range. As a result, a protocol is not adaptive to potential patterns in the mobility of nodes. For example, if two nodes never meet during daytime, the micro-level protocol will still wake-up the nodes during a day, and efficiency therefore will not be optimal.

In this chapter we investigate a machine learning technique to dynamically learn mobility patterns of mobile nodes and wake-up the nodes only when the contact is expected. The work is motivated by several examples of temporal patterns observed in human and also animal mobility traces.

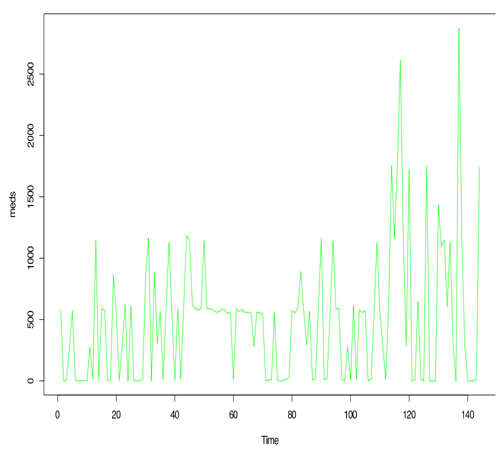
We outline a method for exploiting temporal patterns for efficient node discovery in mobile networks. We first present an approach for partially mobile network, where a fixed wireless infostation tracks the presence of resource constrained mobile nodes. The method uses a simple learning technique to learn and follow diurnal regularities in the connection patterns of links. We then extend an approach for fully mobile networks, where any node (not only a sink) can detect and exploit activity patterns by adaptively tuning its duty cycle.

4.2 Patterns in Mobility

There is a growing evidence in the networking and biological communities that the activity of living creatures follows certain temporal patterns or cycles. These temporal cycles depend on various factors including availability of food, migration and social structure and have been extensively studied in chronobiology [Ebert et al., 1983]. These cycles are linked to solar and lunar related activities and are also called as biological rhythms. Depending on frequency the cycles are categorized into circadian, infradian and ultradian rhythms [Dunlap et al., 2004].



(a) Intel data.



(b) Intel data.

Figure 4.1: The probability is highest during midday. a) The activity starts at 7am reaching peak at 11am, drops sharply after 13pm and reaches the minimum by 16pm. b) The contact durations are more consistent during a particular time of day.

Infradian rhythms are cycles with a duration longer than a day, such as reproduction cycle in some animals. Ultradian rhythms have a period shorter than a day, such as cycling of phases in human sleep or cycles in hormone production.

One of the most important rhythms in chronobiology is a *circadian* rhythm, a 24-hour cycle, followed by most living organisms such as bacteria, plants, animals and humans. To demonstrate how the circadian rhythm affects the network connectivity we conducted a preliminary analysis of both human and animal mobility traces. Figure 4.1 shows a probability of a physical encounter between people in a human mobile network depending on a time of day. There is a strong temporal pattern with a peak at about 11am and a smaller peak after 1pm. The analysis was performed on Intel/Haggle human mobility dataset, where 36 people were tagged with short-range Bluetooth devices logging the starting time and duration of pairwise contacts for over 6 days [Scott et al., 2006]. The analysis has also shown that the nodes spend an average of 60% of time isolated from other nodes with a minimum isolation time of 37%, which means that there is no need for the nodes to stay awake during these intervals. The median duration of encounter falls into 600s - 1200s interval. The median contact duration does not depend strongly on time of day, except a short interval in the second half of the day. Human activity patterns have also been analyzed and used in the context of SAPHE assisted living project by BT [SAPHE-Project, 2008]. The system envisaged in the project captured the human activity

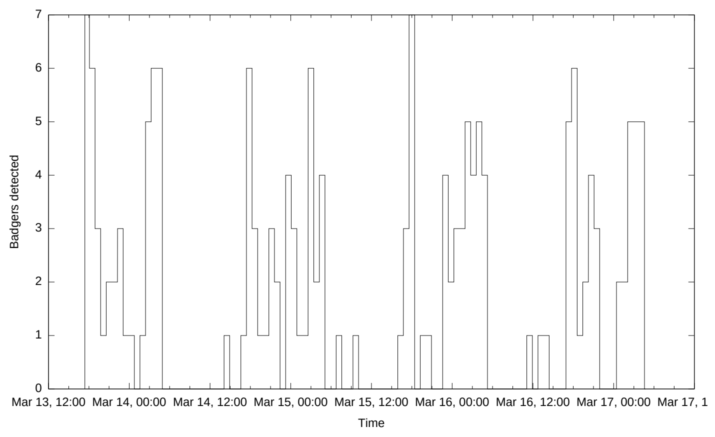


Figure 4.2: Histogram of badger activity

patterns to identify emerging health problems and abnormal situations.

Figure 4.2 shows periodic patterns in the mobility of wildlife badgers. The badgers have been tagged with active RFID tags and their physical presence was registered by multiple RFID readers deployed in the Wytham Woods, Oxford. Figure 4.2 shows the probability of a badger being in a physical range within one of the RFID readers depending on a time of day[1]. It shows a pattern, where badgers are most active during the night and rarely come out during the day. The pattern has two large peaks with several small peaks repeating every day. According to zoologists the activity pattern depends on various factors including the light levels, microclimate and season. This analysis showed examples of strong periodic patterns in human and animal mobility that could be exploited for link scheduling.

In this chapter we investigate whether it is possible to exploit long term periodic patterns in the node mobility. While temporal patterns might be quite complex, we consider the simplest case, when the connection patterns between the nodes have some hourly variations but are stable in the long term. The exploitation of these patterns can be used for a dynamic resource scheduling in a number of scenarios. For example, a wireless node attached to a badger might need to periodically contact a fixed base station located at the burrow entrance to upload collected data. As seen from the example scenario on Figure 4.2 the best time to contact a home base station is at night, when the probability of contacting a base station is the highest. The node can therefore optimise its duty cycle to look for a base station during certain hours of the night, rather than continuously throughout a day. In another scenario, wireless nodes on the badgers need to communicate with each other to relay information to a base station in a multi-hop manner [Vahdat and Becker, 2000]. As the mobility of nodes exhibits a certain pattern the

[1]The data set will be described in more detail in Chapter 7 (Deployment)

nodes could tune their wake up schedule depending on that pattern. These scenarios show that the knowledge of temporal patterns have a potential of reducing the power consumption. The rest of the chapter will contain a description of an approach to exploit these patterns aimed at reducing the duty cycle of the radio.

4.3 Problem Formulation

The goal of the approach is to devise a simple adaptive algorithm to control the scanning rate that reduces the node's power consumption while maintaining the number of successful encounters. The approach is based on the intuition that a high scanning rate will guarantee quick discovery of mobile nodes but will waste energy, especially in situations, where no encounters are likely to occur. On the other hand, a low scanning rate can miss many important contacts.

Intuitively, to detect more encounters, a node needs to sample more frequently when more encounters are expected. Similarly, the node should avoid scanning when no encounters are expected (e.g., during a day, for some animals). Thus it seems natural to formulate the problem as a minimizing the power consumption while maintaining the number of successful encounters:

$$min \sum_i d_i \quad s.t. \quad R = \sum_i E_i * d_i \tag{4.1}$$

Where E_i is the number of detected encounters in timeslot i, d_i a duty cycle at time i. The available energy has to be allocated in such a way, so as to maximize the number of successful encounters. The frequencies of encounters p_i in each timeslot depends on time and are not known in advance.

This formulation of the problem, however, leaves open the question on how the energy budget has to be actually allocated. Let us consider the budget allocation problem in the example scenario.

Budget Allocation Example

Consider example network activity graphs taken and MIT Reality Mining traces (Figure 4.3). These traces were collected using 96 people carrying Bluetooth mobile phones over a duration of 292 days. The phones were programmed to register the opportunistic sightings of other phones and therefore reflect the daily activity patterns of humans over extended period of time. The graphs show the probability of an encounter between any two nodes in the network depending on time of day for each trace respectively. As we see there is a clear pattern, which shows that the encounter probability depends on time of day[2].

[2]More details about each of the traces are available in Section 6.1 (Evaluation).

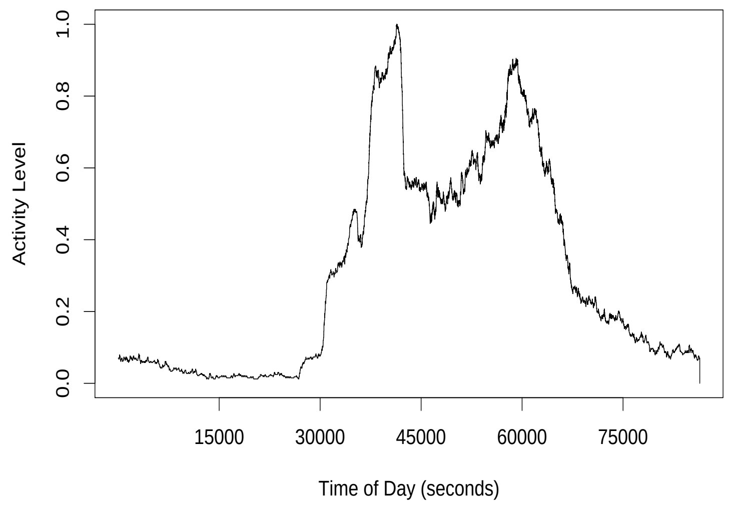


Figure 4.3: Distribution of activity by time (MIT dataset).

To maximize the number of detected encounters, for instance, a mobile node could be trained to wake up between 7am and 5pm (Figure 4.3), when the probability of contact is the highest and, occasionally, try other timeslots to detect any potential changes. Alternatively, the node can pick a timeslot with probability proportional to the height of the peak. Thus, the highest peak would be more likely to be chosen, but if there are several peaks, they will be all equally likely to be chosen. Finally, it could be trained to spread the load to all time slots, but adjusting a duty cycle proportionally to the height of the peak, thus spreading risk evenly throughout a day.

In uncertain environments, exact prediction can be a difficult task, which could be approached by using various methods including time series analysis and other prediction techniques. The major problem with prediction-based techniques is that getting high-resolution data is expensive. In other words, optimal control of a duty cycle requires knowledge about actual node arrival patterns. However, getting this high-resolution data about node arrival patterns requires a node to work at a very high duty cycle. This, in turn, negates the purpose of duty cycle control.

An alternative to prediction-based techniques is *reinforcement learning*. With respect to prediction techniques, reinforcement learning does not try to predict the parameters of the environment, but uses the input from the environment to directly control the actions of an agent [Kaelbling et al., 1996].

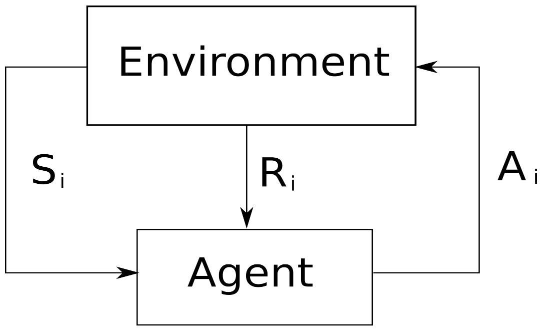


Figure 4.4: Reinforcement learning. The agent perceives an environment through signals S_i, takes an action A_i and receives a reward R_i.

The next section introduces reinforcement learning framework, the model behind the idea, and the possible solution strategies.

4.4 Reinforcement Learning Framework

Reinforcement learning [Kaelbling et al., 1996] provides a formal framework for describing and solving our problem. Reinforcement learning algorithms try to learn by themselves in a trial-and-error fashion by iterating through all possible options and remembering the best ones. A reinforcement learning framework is particularly suitable to our problem because it naturally considers a trade-off between exploration and exploitation phases. It has been successfully applied to problems in operations research, control and robotics. In this section we will present a short overview necessary to understand our approach. A detailed review of the reinforcement problem is given in [Kaelbling et al., 1996] and [Sutton and Barto, 1998].

In Reinforcement learning an agent learns by trial and error by remembering the best actions. The environment is presented as a set of states, $s_t \in S \times T$ and the agent can take a finite number of actions, $a \in A$ (i.e., sleep or stay awake). An agent observes the current state and selects the best response a_i. The environment responds with a certain reward $R : A \times S \rightarrow \Re$, which depends on the current environment state and the agent's action. The goal of an agent is to maximize the reward by learning the optimal policy $p \in P$. The policy is a mapping between states and actions to take and can be calculated using various learning algorithms.

4.4.1 K-armed Bandit Problem

The bandit problem is a classical problem in decision theory originally introduced by [Robbins, 1952]. The bandit problem has received much attention because it models the trade-off between exploration (trying to find the best arm) and exploitation (selecting the arm, which is believed to be the best) in a learning agent. Each arm of the machine when pulled, produces a certain amount of reward, and pulling an arm costs a coin. The goal of the gambler is to identify the best strategy of trying different arms to maximize the total reward.

More formally, a k-armed bandit problem can be described as a set of distributions $B = (R_1, R_2, R_3...R_k)$, where each distribution R_i represents each arm of the bandit. $(\mu_1, \mu_2, \mu_3...\mu_k)$ are the mean values of reward distributions. The goal is to maximize the total reward. The bandit problem is formally equivalent to one-state Markov Decision Process. The crux of the multi-armed bandit problem is that the probability distributions of rewards for each arm have unknown distribution. If a gambler finds an arm, which gives a certain (positive) reward, shall it stick to current arm or try other options at extra cost?

The examples of *k*-armed bandit problem include clinical trials, where several different drugs need to be compared while minimizing the patient loss. A fixed allocation would allocate each drug to a fixed groups of people for the entire duration of a trial. In an adaptive allocation, a drug is allocated depending on the success rate, the patients are switched to better drugs continuously during the trial.

There is a range of solutions to this problem, starting from formally justified techniques to more intuitive ad-hoc techniques. We restrict our techniques on the simple methods, which have shown to perform well in other problems domains.

The formal based techniques include dynamic programming approach, Gittins allocation indices [Gittins, 1979] and learning automata [Sutton and Barto, 1998]. Gittins allocation indices assign an index value to each arm of the bandit. Pulling an arm with a maximum index value provably guarantees a maximum reward. The index values are independent for each arm of the bandit. The learning automata is a method from adaptive control theory and presents a *k*-armed bandit problem as a finite state automaton. The problem representation as finite state machines is not always convenient, although leads to near optimal results. The formal techniques are based on the assumption that the environment is stationary and the probabilities of reward do not change over time. In reality, the environment is not stationary and gradually changes over time.

Ad hoc techniques such as random picking, ϵ-greedy, softmax, optimism in the face of uncertainty and interval based approaches are much more intuitive but are not formally justified. In ϵ-greedy the agent picks an arm with the highest expected reward and tries other arms randomly, with probability ϵ. In softmax, the probability of selecting an arm is proportional to the expected reward. In randomized strategy, which is the most basic strategy, the agent picks an arm randomly. In the optimism under uncertainty technique, the agent optimistically explores possible options until they prove to be negative by experience.

In the following section we describe our problem in terms of *k*-armed bandit problem and investigate the effect of various strategies.

4.4.2 Model

We now introduce the formal model behind the approach. We divide time into d days and each day into N timeslots. At each timeslot t, an agent observes a reward $er(d,t)$, which is a number of encounters in timeslot t and adjusts its duty cycle $DC(d,t)$ accordingly.

The number of encounters at each timeslot is estimated using an exponentially weighted moving average (EWMA) filter by taking into account the measurements from previous days:

$$er(d,t) = r(d,t) * \alpha + er(d-1,t) * (1-\alpha) \tag{4.2}$$

Where $r(d,t)$ is the number of encounters observed in the current day. $er(d-1,t)$ is the previous value of the number of estimated encounters. The weight assigned to past measurements $(1-\alpha)$ depends on how responsive the node has to be to changes in node arrival patterns. High values of α put more weight to recent measurements and make an algorithm more responsive to seasonal changes. Low values of α make an algorithm more robust to noise, but less responsive to seasonal changes.

The duty cycle at timeslot t is proportional to the expected number of encounters in that timeslot:

$$DC(d,t) = DC_{max} \frac{er(d,t)}{\sum_{i=1}^{N} er(d,i)} \tag{4.3}$$

Where DC_{max} is a target duty cycle. If there are several peak hours during a day, the budget will be spread evenly among all peaks. During quiet times the node continues to sample the environment but with lower frequency. If a value of a duty cycle for a timeslot is below a certain threshold, it keeps exploring that timeslot with a small fraction of the target duty cycle DC_{max}. On the first day, when information about expected encounters is unavailable, the duty cycle is spread uniformly across all timeslots. Our evaluation will show how effective this technique is with respect to static or random approaches.

Learning the environment accurately would require the node to stay awake all the time, which contradicts the goal of saving energy, in other words, learning is expensive. The approach we develop, balances the exploration (learning) and exploitation costs. We now describe the strategies that could be used to adapt the node's duty cycle and a random strategy, which we will use as baseline for the evaluation (Chapter 6).

4.4.3 Strategies

4.4.3.1 Random

In a random strategy (which we use for comparison) the node spreads its energy budget evenly throughout a day, i.e., it sends beacons with a certain fixed interval. The strategy is equivalent to normal asynchronous wake-up scheduling with fixed duty cycle, so would not require additional implementation. The obvious problem with the random strategy is that a node will waste resources when there are no nodes around. This will become evident in our evaluation.

4.4.3.2 Greedy

In a pure greedy strategy, a node always selects the best *known* timeslot. During this timeslot a node spends all its daily budget. The problem with a greedy strategy is that it can converge too early on suboptimal solution [Kaelbling et al., 1996] without finding for an optimal solution.

ε-Greedy is a popular and simple strategy for solving the k-armed bandit problems was first introduced by [Robbins, 1952]. In this strategy, a node selects the greedy action, but also explores other choices with a probability ε. This guarantees that the nodes always invest a certain amount of resources in learning about the environment. A variation of this strategy, called GreedyMix [Cesa-Bianchi and Fischer, 1998], starts with relatively high ε and decreases focusing on exploitation phase as the agent gathers more information about the environment. Other variations of ε-greedy strategy exist but we focus on the standard ε-greedy to cope with slowly changing environment.

4.4.3.3 Boltzmann exploration

In Boltzmann exploration the selection of a timeslot *a* depends on its expected reward *ER(a)*. The advantage is that better timeslots have higher chances of being explored. The reward has the following distribution:

$$P(a) = \frac{e^{er(d,t)})/T}{\sum_{a \in A} e^{er(d,t)/T}} \tag{4.4}$$

The probability of choosing an action *a* is proportional to exponent raised to the power of expected reward *ER(a)* normalized by sum of exponents over all actions. The Boltzmann exploration is a probability matching method, because it chooses an arm depending on the optimality of the arm.

The parameter T is referred to as temperature of the system (in a metaphorical way). The temperature parameter sets a trade-off between exploitation and exploration. The low temperature values will encourage greedy behaviour, whereas high temperature will force more exploration. Also known as *softmax*, the probability of choosing will depend on the past history. A variation of Boltzmann exploration with decreasing temperature exists, which starts with higher exploration and focuses on exploitation phase [Cesa-Bianchi and Fischer, 1998].

4.4.3.4 Balanced

The balanced strategy is based on the idea that the node should not bet on any single arm, but spread its risk to all arms depending on the arm's optimality. In a balanced strategy we propose to dynamically adjust the node's duty cycle proportionally to an expected reward. Therefore the node *does not commit* to any timeslot, but spreads its energy proportionally to the expected reward. The node sets its duty cycle according the following rule:

$$DC(d,t) = DC_{max} \frac{er(d,t)}{\sum_{i=1}^{N} er(d,i)} \tag{4.5}$$

In the next two subsections we will apply our model to two scenarios progressively more mobile. The first contains a fixed base station, which is monitoring several mobile objects and the second contains only mobile objects.

4.5 Node Discovery Protocol for Partially Mobile Networks

In this section we present an algorithm for adaptive node discovery, which uses the presented approach. The daily budget assignment could be performed by the application, depending on the known energy availability: for example in the scenario we envisage, it is very clear how big the batteries can be and how long the zoologists want them to last for, therefore the daily budget can be inferred.

Application Scenario

The scenario in the introduction chapter described a network consisting of mobile tags attached to wildlife animals. In that scenario, the small animals, such as badgers are tagged with small sensors, which collect environmental information. The sensor nodes are generating readings periodically and store it in a local flash storage. The goal was to collect all information from the tags in the most energy efficient way.

In this subsection, we simplify this scenario by relying on fixed sinks to detect and collect information from the mobile tags. A number of static base stations (sinks) nodes are positioned in the forest for data collection when badgers come in communication range (Figure 4.5). Biologists gather the data from the sinks every now and then while passing through the forest. This scenario captures the typical requirements present in many wildlife tracking applications. Similar requirements, for example exist in Autonomous Habitat Monitoring project [Naumowicz et al., 2008], where a sink needs to detect tagged birds in communication range to download data.

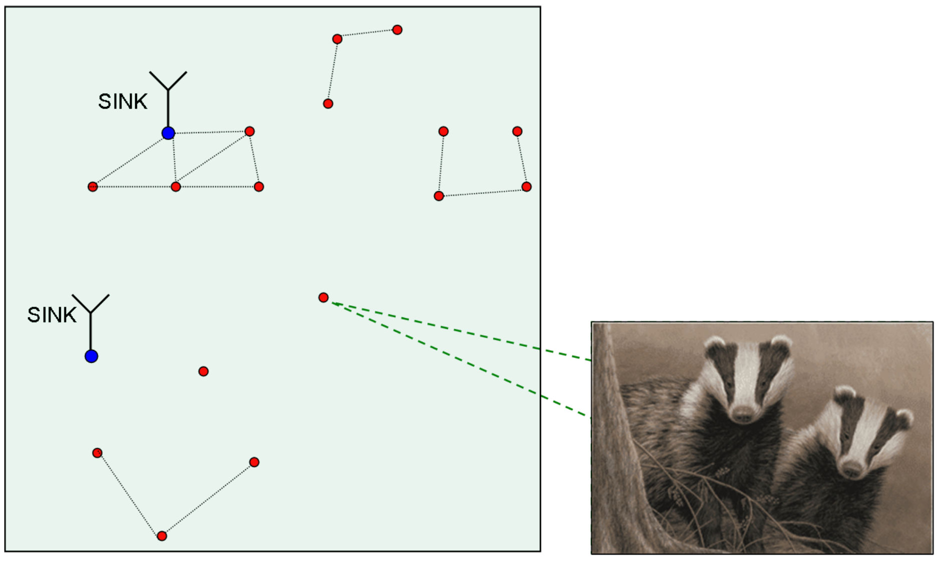


Figure 4.5: Example scenario. The sensors are used to track badgers in the forest.

By making this simplification, we shift the power consumption from the resource constrained tags to the fixed sinks. The important issue in this scenario is that badgers are quite small animals so the tags cannot be very large. As animal trapping and tagging is a labour intensive activity the sensors should be made to last as much as possible. The sinks have larger but also limited batteries.

We now describe a method for duty cycling sinks, while enabling them to detect and communicate with the tags in the most efficient way.

4.6 Protocol

1. The sink starts by following a random strategy, i.e., spreads its duty cycle equally in each timeslot. As it discovers new nodes it dynamically readjusts its budget according to the following steps.

2. Once discovered, the nodes remain synchronized for the duration of an encounter. Short term synchronization is possible with built-in timers without the need of globally synchronized clocks. As long as there is at least one node in range, the node sends periodic keep-alive messages every $T_{keepalive}$ seconds.

3. If a node does not hear from a neighbour for T_{expire} seconds, it assumes an encounter is terminated and increments the timeslot counter.

   $C_t = C_t + 1.$

4. At the end of each day a node updates its timeslot counters

$M_t = C_t * \alpha + M_t * (1 - \alpha), t = 0..N_{slots}.$

Where M_t is an estimated encounter frequency at timeslot *t* and C_t is the actual number of encounters in timeslot *t* registered during current day. The node then resets the daily counters C_t.

5. At the beginning of the current timeslot *(t)*, a node sets a beacon rate to be:

   $F_{beacon} = \frac{M_t}{\sum_i M_i} \frac{B}{E_{beacon}}.$

   Where *B* is a daily energy budget, E_{beacon} is an amount of energy required to scan the neighbourhood. The node converts the beacon frequency into interval time between beacons

   $T_{beacon} = 1/F_{beacon}.$

   If the duration of this period is longer than the timeslot duration $T_{timeslot}$, the node beacons with a probability $p = \frac{T_{beacon}}{T_{timeslot}}$. The node then schedules the next wake up by the beginning of the next timeslot. The node has preconfigured minimum and maximum beacon rates F_{min} and F_{max}. The minimum beacon rate is needed to guarantee a certain level of exploration, even when no discovery is expected. The maximum beacon rate limits the amount of energy a node spends in one timeslot.

The protocol will be evaluated in the Evaluation (Chapter 5).

4.7 Fully Mobile Extension

In this section, we describe the application of an approach to multi-hop wireless sensor networks, where all the mobile nodes need to discover each other. This could be useful in a number of situations, where placing of static sinks is not desirable or possible. Some animals, such as seals, do not have fixed habitat and migrate long distances depending on season. All nodes are assumed to have the same power budget.

4.7.1 Basic Version

In the basic version each node follows an adaptive (balanced) duty cycling strategy developed in the previous section. Each node monitors the surrounding activity and adjusts its duty cycle to the busiest timeslots.

4.7.2 Experimental Version

In the basic version the node adapts its duty cycling depending on the surrounding activity. In the experimental version the node adapts its duty cycling towards meeting specific nodes in the network. For example, the nodes might tune their duty cycling towards meeting a sink only. In this case, they have observe the sink's activity as opposed to overall surrounding activity. In another example, the nodes from different social groups might tune their duty cycling towards their respective group leaders only. The latter option actively changes the network structure by keeping the frequent or regular communication links and excluding less frequent or regular links between the group members.

The experimental version keeps statistics for *each individual* node in the network. A node then aggregates the statistics of all nodes it is interested in meeting and calculates a duty cycle using a balanced strategy. Thus, the steps 3, 4 and 5 are different from the basic version.

1. The node starts by following a random strategy, i.e., spreads its duty cycle equally in each timeslot. As it discovers new nodes it dynamically reajusts its budget according to the following steps.

2. Once discovered, the nodes remain synchronized for a duration of an encounter. Short term synchronization is possible with built-in timers without the need of globally synchronized clocks. As long as there is at least one node in range, the node sends periodic keep-alive messages every $T_{keepalive}$ seconds.

3. If a node does not hear from a neighbour for T_{expire} seconds, it assumes an encounter is terminated and increments the timeslot counter. The node keeps counters for each individual node in the network.

   $C_{i,t} = C_{i,t} + 1.$

4. At the end of each day a node updates its timeslot counters

   $M_{i,t} = C_{i,t} * \alpha + M_{i,t} * (1 - \alpha), t = 0..N_{slots}.$

   Where $M_{i,t}$ is an estimated encounter frequency at timeslot *t* and $C_{i,t}$ is the actual number of encounters in timeslot *t* registered during current day. The node then resets the daily counters $C_{i,t}$.

5. At the beginning of the current timeslot *(t)*, a node sets a beacon rate to be:

   $F_{beacon} = \frac{M_{i \in S,t}}{\sum_i M_i} \frac{B}{E_{beacon}}.$

   When calculating the beacon frequency, the node only aggregates the statics of the nodes it is interested in meeting ($i \in S$).

   $T_{beacon} = 1/F_{beacon}.$

4.8 Interaction with Micro-level Protocol

In this section we describe the interaction of macro and micro level protocols. The adaptive duty cycling is applied as adaptive parameter changing for MAC*ron* protocol, namely changing the beacon frequency of underlying medium access protocol. We now describe the configuration of lower level micro-level protocol for partially mobile and mobile scenarios.

In the partially mobile scenario, the main goal of MAC configuration is to shift the communication load from resource constrained tags to more resource rich static sinks. Specifically, we show the MAC*ron* configuration for this setting. The MAC*ron* protocol introduced in Chapter 3, duty cycles the nodes asynchronously, independently of each other. In the default configuration, all nodes are allowed to beacon and therefore consume energy for node discovery. In our scenario, given that mobile nodes have to operate for as long as possible, we configure the mobile nodes to operate at the lowest possible duty cycle, where the only task they perform is periodic carrier sensing. The lower bound of tag's duty cycle is defined by the carrier sensing check interval:

$$D_{mobile} = \frac{T_{carrier\_sampling}}{T_{check\_interval}}$$

At $T_{check\_interval}$ = 3s, D_{min} is 0.3%, providing theoretically 4.7 month of operation on a coin battery[3]. With this configuration, the only way to increase tag's lifetime is to either increase check interval, or use larger battery, which is undesirable as it would increase the size and weight of the tag.

The sinks are configured to send beacons to advertise their locations and discover mobile tags. The beacon transmission (scanning) has to be long enough to wake-up a mobile nodes with communication range. Mobile nodes have to reply to a beacon by sending an acknowledgment. The lower bound of sinks' duty cycle is defined by the beacon frequency. The sink's duty cycle is therefore:

$$D_{sink} = \frac{t_{check\_interval}}{T\_beacon\_interval}$$

At $T_{beacon\_interval}$ = 300s, the sink's duty cycle is 1.2% and the approximate lifetime is 3.7 months on AA battery[4].

It is apparent that the sink lifetime is also limited because the communication load is shifted to the sinks, even though the sinks have larger batteries. Taking into account large number of sinks in the forest, it becomes important to optimize the sink's energy consumption.

[3]CR2450 coin battery, 3V, 610mAh.

[4]Alkaline battery, 2x1.5V, 2100mAh.

4.9 Implementation

We implemented the presented protocols in TinyOS-2 [Levis, 2006]. The protocol for partially-mobile networks has been implemented to control the power of RFID readers connected to sensor nodes. The protocol adaptively controls the power to energy intensive RFID readers and controls the amount of awake and sleep time depending on activity. The protocol for fully-mobile networks has been implemented to adaptively control the scanning frequency of the short range radio on the sensor node. More details on the RFID related implementation as well as the system overview will be presented in Chapter 7.

The implementation uses a discovery module, which keeps daily statistics of encounters and consults the above application in order to decide on the optimal duty cycle for a current timeslot. The implementation takes 16232 bytes of ROM and 2016 bytes of RAM (not counting message arrays and micro-level implementation).

The node stores statistics in a *uint32_t model[TIMESLOTS]* array and current daily statistics in a separate *uint32_t c* array. TIMESLOTS parameter is defined at compile time. The application sets a daily budget using a *setBudget(uint32_t budget)* call during initialization procedure. Every time a node discovers or hears from another node it calls an *encounter()* function, which logs the time of contact.

An application can request a recommended duty cycle by calling *getDutyCycle()* function. A duty cycle indicates the proportion of daily budget an application can spend in the current timeslot. An application then converts the recommended duty cycle value into a discovery beacon period. If the resulting beacon period is longer than the timeslot duration, the node beacons with a probability proportional to beacon period and schedules next wake up at the beginning of next timeslot.

Table 4.1: Duty cycle API.

Function	**Description**
`void reset`	reset model
`void setBudget(uint16_t)`	sets daily budget
`uint16_t getBudget()`	retrieves a daily budget
`void encounter()`	registers a successful encounter
`uint16_t getDutyCycle()`	get recommended duty cycle
based on the model	

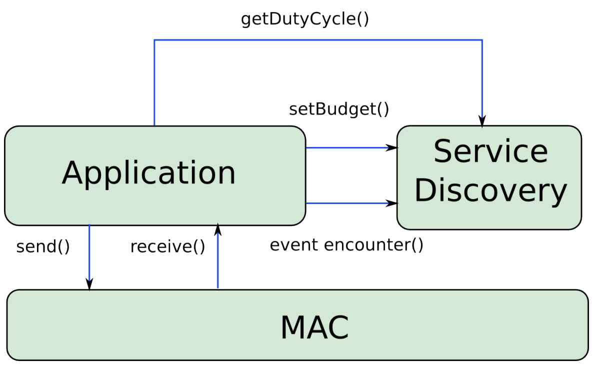


Figure 4.6: Implementation architecture.

4.10 Related Work

This section contains relevant work on application of machine learning in sensor networks.

In this subsection, we review the state of the art in using machine learning and stochastic methods for wireless sensor networks. There is a growing interest in applying machine learning techniques for sensor network applications. The machine learning techniques can be applied for high-level information extraction, which can be done locally, without sending the data to the base station. Since many real world phenomena, such as temperature or humidity are spatially and temporally correlated, there is a potential for reducing the raw data traffic by using machine learning techniques, which observe patterns and only report changes in the environment. [Chu et al., 2006] propose a data collection technique, which exploits spatial correlations between the nodes in the network. The nodes use relatively simple models to detect and report changes in the environment. [Herbert et al., 2007] propose an information collection technique using an ARIMA time series model [Box and Jenkins, 1994] to exploits temporal correlations in the environment. In the approach, both sink and deployed sensors model incoming sensor data using an ARIMA model, and communicate only if the new data does not match the prediction, which reduces data traffic in the network. The performance of these approaches depend on the selection of the correct model, which is able to describe the data and correctly parameterizing this model. Both model selection and parameterizing tasks are computationally complex and require estimating the model against a variety of estimators.

Machine learning techniques have been used to detect outliers in the raw streams of sensor data. The problem arises when data contains missing or extreme values, in which case the sensor needs to recognize whether the extreme value occured due to a fault in the equipment or structural change in the environment, which needs to be reported. The outlier detection is typically used to either suppress such extreme values or detect them. In the first case, the outlier detection methods help clean the data due to hardware faults, the second helps detect unusual or rare events in the massive amount of data and is used in intrusion detection systems, anomaly detection and diagnostic systems. As outliers are very unusual events, statistical generalizing is hard or impossible, therefore most methods rely on heuristics. These heuristics fall into unsupervised and supervised category and are usually done offline, on a centralized powerful work station. Machine learning tehniques have been used to move this task onto sensors, to reduce data traffic and make sensors report unusual events only. [Branch et al., 2006] develop an in-network method for outlier detection in wireless sensor network. The sensors are automatically generating a model based on the data and sends the model parameters instead of raw data to a gateway. The incoming data queries are executed by the gateway and run against the statistical model.

Machine learning has also been applied for a variety of wireless sensor network tasks such as link quality prediction [Wang et al., 2005], routing optimization [Wang et al., 2006], broadcast optimization [Colagrosso, 2007], MAC protocols [Liu and Elhanany, 2006][Stone and Colagrosso, 2007], localization and location tracking [Pan et al., 2007]. Wireless sensors are typically very resource constrained, in terms of operating memory and processing power, which means the applications of machine learning has to be restricted to very simple algorithms.

[Jain et al., 2004] use oracles in their delay-tolerant networking (DTN) framework to aggregate statistics of contacts. [Jun et al., 2006] use oracles for hierarchical power management in DTN nodes. In this protocol, we use an oracle, which identifies and uses temporal patterns in the node encounters. We assume that the temporal patterns are stationary or slowly varying.

Mainland et al. [2005] propose a machine learning based approach for adaptive resource allocation for sensor networks. The approach dynamically schedules sensor activity. The sensors are modeled as self-interested agents that try to maximize their profit and a simulation based evaluation is presented. We similarly adjust the detection nodes schedules dynamically, however, the nodes are learning and exploiting long-term periodic patterns to spread daily budget according to expected activity. We also integrate this in an heterogeneous network deployment and present a real evaluation of the system.

[Hsu et al., 2006] use a simple learning approach to adapt node's duty cycling for energy harvesting systems. They note that the energy harvesting system, such as solar battery have a temporal profile, which reaches a peak during the day and drops to zero at night. They adapt node's duty cycling for this temporal profile, so that the node does not consume more energy than was generated during the day.

In [Wang et al., 2006] the offline learning algorithm is used to learn link quality information and optimize routing protocols. The authors then propose a new protocol, which uses link quality metrics calculated using a learning algorithm. The authors state that the new link quality metrics works in situations where ETX approach fails. BoostMAC [Stone and Colagrosso, 2007] uses learning techniques to learn the wake up times of neighbouring nodes to reduce the expensive extended preamble duration. The idea is similar to [El-Hoiydi and Decotignie, 2004], which also used simple learning technique to predict the wake-up times of neighbour nodes.

4.11 Summary

We have presented a macro-level approach for adaptive duty cycling in mobile wireless sensor networks. We first have demonstrated the presence of temporal cycles in the human and animal mobility by analyzing the real mobility traces from both publicly available and our own data sets. The examples show that the probability of a node meeting another node strongly depends on a time of day, a fact, which we exploit for dynamic duty cycling. We then proposed an approach, which learns the temporal patterns in the node mobility and uses this information to dynamically schedule the power budget of nodes. Under this approach, the node spends most energy only when the encounters are expected (*exploitation*) and sleeps the rest of the time. To make an approach adaptive for potential changes in mobility pattern, an approach also spends a certain amount of resources into *exploration* of inactive time slots. The approach uses simple techniques from reinforcement learning to balance between exploration and exploitation of the environment. Finally, we have shown through implementation on real Tmote sky nodes that the approach is suitable for resource constrained sensor nodes. We report the evaluation of the protocol in Chapter 6 and Chapter 7 (Deployment).

Chapter 5

Evaluation of Micro-level Scheduling

We have now described our two layer duty cycling approach. We now evaluate the performance of the different layers with a mixture of real testbed measurements and simulations.

In this section we evaluate MAC*ron* protocol described in Chapter 3.2 experimentally through measurements on a real testbed. Our aim is to demonstrate the protocol performance in real settings for typical mobile wireless sensor network applications such as data transfer and data dissemination. We present the results of measurements in a small indoor testbed consisting of 6 Tmote Sky sensor nodes for unicast and broadcast data transmission.

5.1 Experimental Setup

We have tested our implementation in an indoor testbed consisting of 6 Tmote Sky [MoteIV, 2006] sensor nodes. Tmote Sky sensor nodes are based on low power 8Mhz TI MSP430 microcontroller and contain 10kB RAM, 48kB of program flash and 1MB of data memory. TI MSP 430 is an ultra-low power 16-bit RISC microcontroller with a processing performance of 16 RISC MIPS. The microcontroller has 6 low power modes 5.1 with power consumption ranging from 340uA in fully active mode (AM) to 0.1uA in sleep mode (LPM4). Depending on the current LPM mode the microcontroller can be woken up by a timer (LPM-X) or an external interrupt (LPM4). The operating system puts the microcontroller to sleep automatically when the task queue becomes empty. There is no explicit command in the implementation to put the microcontroller into low power mode. The radio as well as other components have to be turned off explicitly.

The sensor nodes are equipped with CC2420 802.15.4 [TexasInstruments, 2008] compatible packet-based radio transceiver operating in 2.4Ghz ISM band. The radio uses a direct sequence spread spectrum (DSSS) radio and provides a transmission rate of 250kbs. The radio has four main states: transmitting, receiving, idle and power down. The energy consumption of all four states are presented in Table 5.2.

Table 5.1: MSP430 Operating Modes

Mode	CPU	MCLCK	SMCLK	ACLK	DCO	Current at 1Mhz, 3.3V
AM	ON	ON	ON	ON	ON	340uA
LPM0	OFF	OFF	ON	ON	ON	70uA
LPM1	OFF	OFF	ON	ON	OFF	-
LPM2	OFF	OFF	OFF	ON	OFF	17uA
LPM3	OFF	OFF	OFF	ON	OFF	2uA
LPM4	OFF	OFF	OFF	OFF	OFF	0.1uA

Table 5.2: CC2420 power consumption

State	**Power Consumption**
transmit, P = 0dBm (max)	17.4mA
transmit, P = -25dBm (min)	8.5mA
receive	19.7mA
idle mode	426μA
power down	20μA

5.1.1 Description of a Protocol Used for Comparison

Even though there is a plethora of MAC protocols for wireless sensor network, there is no single protocol accepted as standard. This is in contrast to, for example, Wi-Fi networks, where 802.11 is a universal standard for many types of applications. The energy-efficient MAC protocols have to minimize the awake time of the nodes. As we argued, uncertainty is a major factor for higher power consumption. The uncertainty of packet arrival, for example, makes the node to wake up periodically and check for a transmission in a channel. The cross layer integration, essentially means that application or domain knowledge (such as topology, latency or traffic patterns) can be used to reduce uncertainty at the MAC layer by incorporating this knowledge into the MAC layer in the form of rules. Eventually, this means that in scenarios, where performance is important, the MAC protocol has to be tightly integrated and dependent upon specific application requirements.

As already mentioned in Chapter 3, among the number of protocols suitable for mobile scenarios we can identify BMAC [Polastre et al., 2004] and X-MAC [Buettner et al., 2006]. BMAC is essentially a low power listening protocol with a flexible API on top. The API allows to change the protocol parameters such as enabling and disabling of acknowledgements, CCA modes, setting CSMA backoffs and carrier check intervals and preamble lengths. X-MAC is a more recent protocol, which uses a number of optimization techniques. We have selected

X-MAC for comparison as we consider it one of the most efficient.

X-MAC uses a sequence of short packets sent back-to-back to wake up a receiver. Upon discovering a transmission a receiver sends an ack to the sender causing it to stop transmission, thus saving both sender's and receiver's energy. The X-MAC also uses an address recognition function, so the nodes do not spend energy overhearing transmissions destined to other nodes. These features allow X-MAC to keep duty cycle low even in densely packed environments. The weakness of X-MAC is that sender's duty cycle is relatively high as it still needs to transmit on average half a preamble to wake up a receiver. Another weakness is that it is not suited to broadcast applications as the transmitter is always stopped by the node, which first detects a transmission. The original X-MAC protocol was implemented for MantisOS operating system [Bhatti et al., January 2005]. We implemented X-MAC for TinyOS for Tmote Sky sensor nodes and used it for comparison with our protocol. The protocol was implemented using hardware acknowledgements supported by CC2420 radio.

We do not compare with the default MAC protocol implemented in TinyOS. The default MAC is a contention based protocol and does not include any power saving mechanisms. The power consumption of default MAC is always at the constant level regardless of existence or amount of the communication traffic.

5.1.2 The Protocol Parameters

The nodes were configured to turn on the radio for an average of 7.2ms, which includes the time to stabilize an oscillator, enable the radio and evaluate RSSI. The RSSI is evaluated by averaging it over 8 symbol periods (128uS) by CC2420 radio. The default interval between carrier checks was set to 784ms and was the same for both protocols. The minimum duration a preamble was set to compensate for a maximum residual error in skew estimation algorithm. This error was calculated based on measurements described in Section 5.4.

While duty cycling, the radio was put into power down mode (See Table 5.1) between carrier check intervals. The startup time from power down mode is longer than in idle mode, due to the need to enable voltage regulator and stabilize an oscillator but it has the lowest power consumption. To avoid potential interference with other sources of 2.4Ghz band traffic such as 802.11 and Bluetooth the measurements were done on channel 26, which does not overlap with 802.11b/g/n spectrum[Petrova et al., 2007]. The transciever power level was set to a maximum of 0dBm.

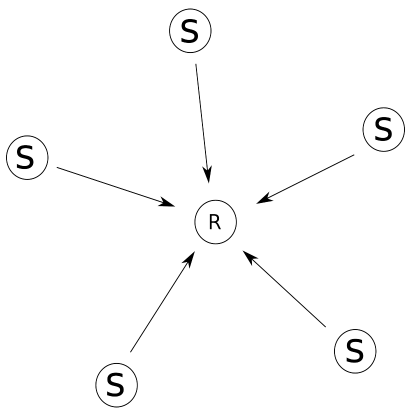


Figure 5.1: Network topology.

5.1.3 Metrics

The primary metric for comparison was the power consumption used for radio communication. We ignore the power consumption related to CPU, Flash memory and other components, as we focus on the power consumption of the radio only. As shown by previous research, radio accounts for the largest share of total energy consumption in wireless monitoring applications [Sohrabi et al., 2000, Pottie and Kaiser, 2000, Doherty et al., 2001, Raghunathan et al., 2002].

We measure the effective node duty cycle as percentage of time the radio spends in active state. The total time in active state consisted of channel sampling, receiving and transmitting the messages. It also includes the time needed to turn on and stablize an oscillator. The node keeps track of total active time by maintaining a special 32 bit variable. The variable is updated every time the node disables the radio and reported via serial port at the end of an experiment.

5.2 Unicast Transmission Experiment

All the nodes were arranged in a star topology with a receiver in the centre (Figure 5.1). For this experiment we generated a light traffic, typical for long-lived applications. Each node was configured to send 80 byte packets to a receiver every 5 seconds. To avoid collisions the nodes were booted at random times within 10 seconds. The experiment duration was set to 180 seconds, which means that each sender node generated approximately 36 messages.

To evaluate the protocol behaviour for different network densities we repeated the experiment for number of senders from 1 to 5. The nodes keep track of their total active time and log the results through USB port to a notebook at the end of each experiment. To avoid resource conflicts over UART serial and radio module, which share the same bus, the nodes reported results after the experiment finished, upon completely turning off the radio.

The result of experiment is shown on Figure 5.2a. As expected, the senders' duty cycle

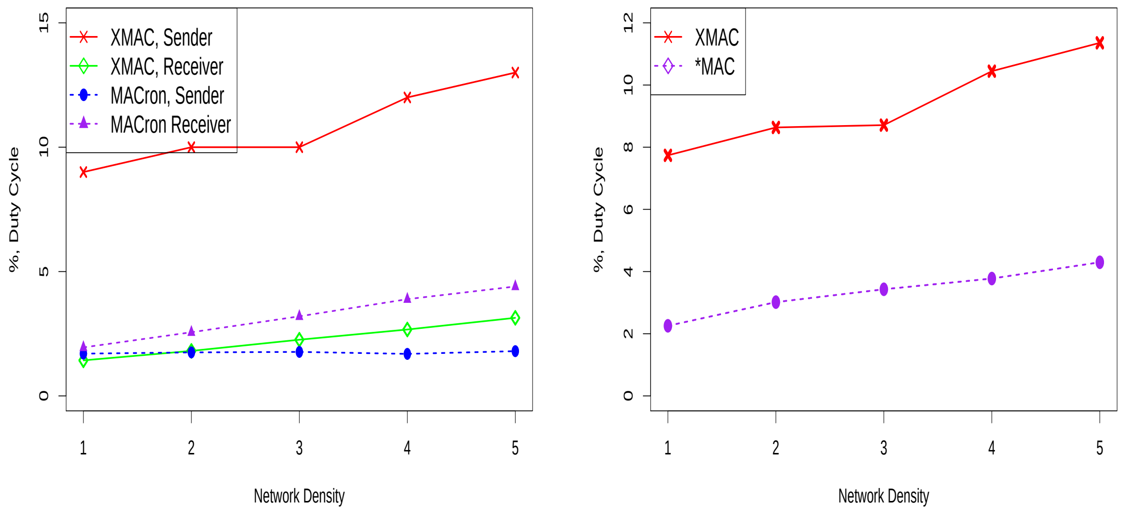


Figure 5.2: a) Duty cycles of senders and receivers versus network density b) Average duty cycle versus network density.

for MAC*ron* is significantly lower than that of X-MAC. Even though X-MAC receiver has the ability to truncate the preamble by sending an ack, the senders still need to transmit on average half a preamble to wake up a receiver. A MAC*ron* sender uses very short preambles to send packets to the receiver, thus saving both senders and receiver's energy. The sender's duty cycle grows slightly with increased network density because of occasional collisions. The receiver's power consumption for MAC*ron* protocol grows slowly with the network density. The power consumption increases due to higher number of received packets.

Figure 5.2b compares the average power consumption of MAC*ron* and X-MAC. The power consumption of MAC*ron* is significantly lower over the entire range of network densities. The nodes in MAC*ron* require up to 4 times less energy than X-MAC to transmit the same amount of data. The experiment demonstrates that the use of on-demand synchronization allows to cut transmission costs.

5.3 Broadcast Transmission Experiment

In the previous experiment, we measured the power consumption for unicast traffic. Many protocols in delay tolerant networks use broadcast communication to simultaneously send packets to all neighbours. Broadcast communication is useful in opportunistic networks, when the nodes need to disseminate the packet to all nodes regardless of an actual address. Periodic

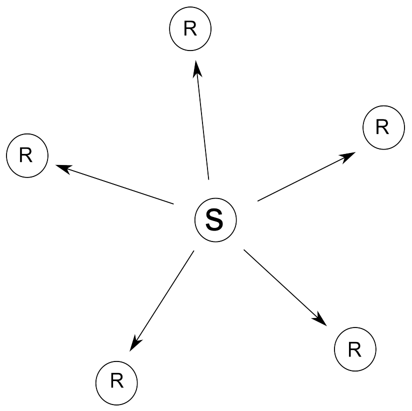


Figure 5.3: Network topology for broadcast application.

broadcasting is used frequently, for example, by many routing protocols [Vahdat and Becker, 2000][Lindgren et al., 2003][Perkins and Royer, 1997] to distribute the link state information, routing tables or presence updates.

In this experiment, we evaluated the effectiveness of preamble tracking feature for a simple broadcast application. The MAC protocol has to be optimized for a broadcast traffic and keep energy consumption low in high network densities. As in the previous experiment, the nodes were arranged in a star topology but with a sender in the centre (Figure 5.3). The sender transmitted a broadcast message to all nodes every 10 seconds.

5.3.1 Sender's Power Consumption

Figure 5.4a shows the duty cycle of sender and receiver nodes for X-MAC and MAC*ron* protocols. MAC*ron* sender synchronizes the nodes at the beginning of experiment and then uses very short preambles to transmit packets. The receivers were configured to respond to sender with acks.

X-MAC's sender is stopped early by the receiver preventing the rest of the nodes from receiving the broadcast message. To succesfully transmit a broadcast message, a sender has to transmit the same message individually to each recepient. Thus, the power consumption grows linearly with network density. An alternative solution is to modify the X-MAC protocol to keep transmitting preamble for the entire duration of check interval to wake up all nodes at once. This would make the sender's power stay constant, but would increase the receivers' power consumption leading to a similar result.

5.3.2 Receiver's Power Consumption

The receivers showed the same duty cycle for both protocols. Receivers in X-MAC send an ack as soon as they detect a transmission, without receiving an entire preamble. The receivers in

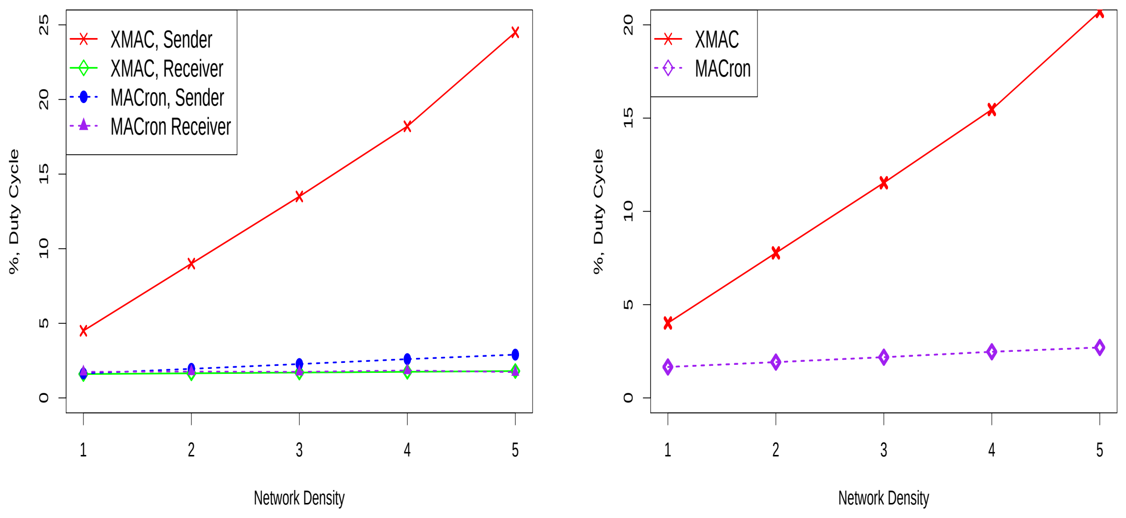


Figure 5.4: a) Duty cycle of senders and receivers in a broadcast application b) Average duty cycle in a broadcast application.

MAC*ron* receive very short synchronized preambles, which keeps their duty cycle very low.

5.3.3 Average Power Consumption

Figure 5.4b compares the average duty cycle of both protocols for broadcast application, which demonstrates that MAC*ron* has power consumption at least 4 times lower than X-MAC for the similar traffic.

The timestamp and schedule information impose a slight overhead on each packet (6 bytes). To increase precision, the packet timestamp is updated at the very last moment, when the radio just begins transmission of a packet from the CC2420 transmission buffer. The field has to be located at the end of the packet and there is a minimum packet length, so that transmission buffer does not underflow before the timestamp is written. The timestamping each packet reduced the time uncertainty by 6 ms, which is the time it takes to transmit a packet from RAM to CC2420 chip and depends on packet size.

5.4 Clock Skew Estimation

To evaluate the efficiency of the skew estimation algorithm (Section 3.2) we performed the measurements to characterize the clock drift of the sensor nodes. The sensor node clocks are sourced by default from internal MSP430 digital oscillator working at 8Mhz frequency. The crystal frequency depends on the temperature, which is the result of clock offsets and skews for

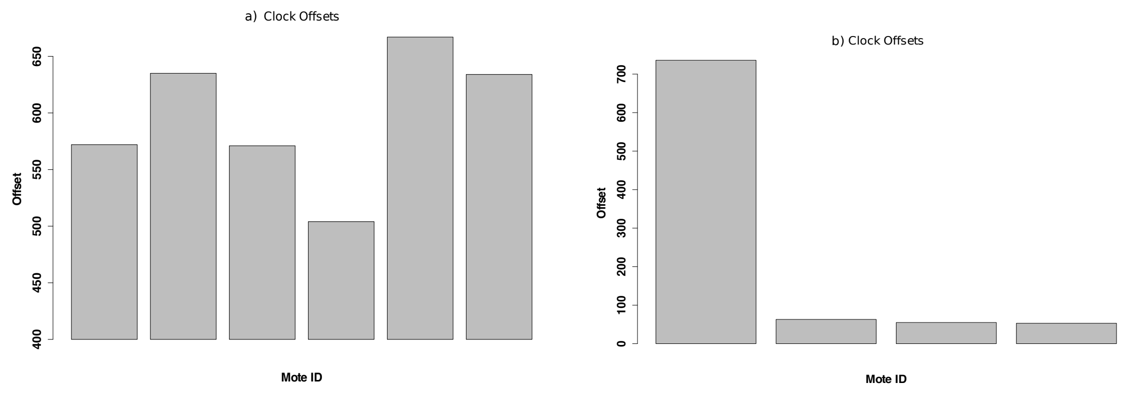


Figure 5.5: Clock offset measurement: a) indoors b) outdoors.

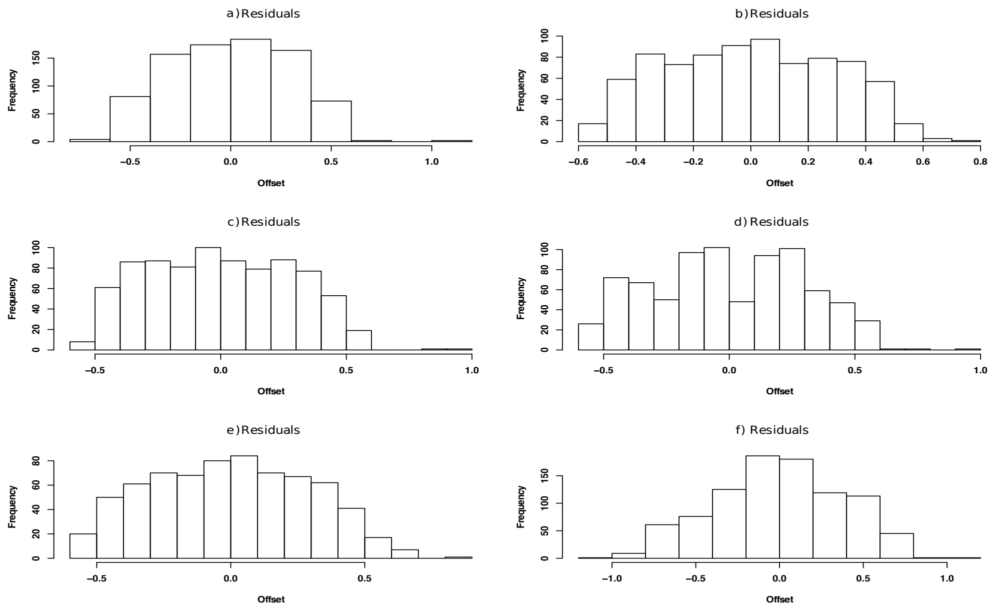


Figure 5.6: Drift measurement for sensor nodes indoors.

each node. In the experiment, all the sensor nodes were connected to a USB hub and were sending packets to a randomly selected sensor node for an hour every 5 seconds. The experiment was first conducted indoors at a room temperature of 24°C. The experiment was repeated ourdoors at temperature 1°C. The temperature remained stable throughout each experiment. The receiving node logged local and remote timestamps via serial interface to a workstation, which logged the data to a text file. The dataset was then imported into R package, which calculated the linear regression coefficients, average clock drift and residual errors.

Figure 5.5a shows the clock offsets between the sensor nodes over an hour. It shows that the clock skew became as large as 650ms, which turned out to be much higher than documented

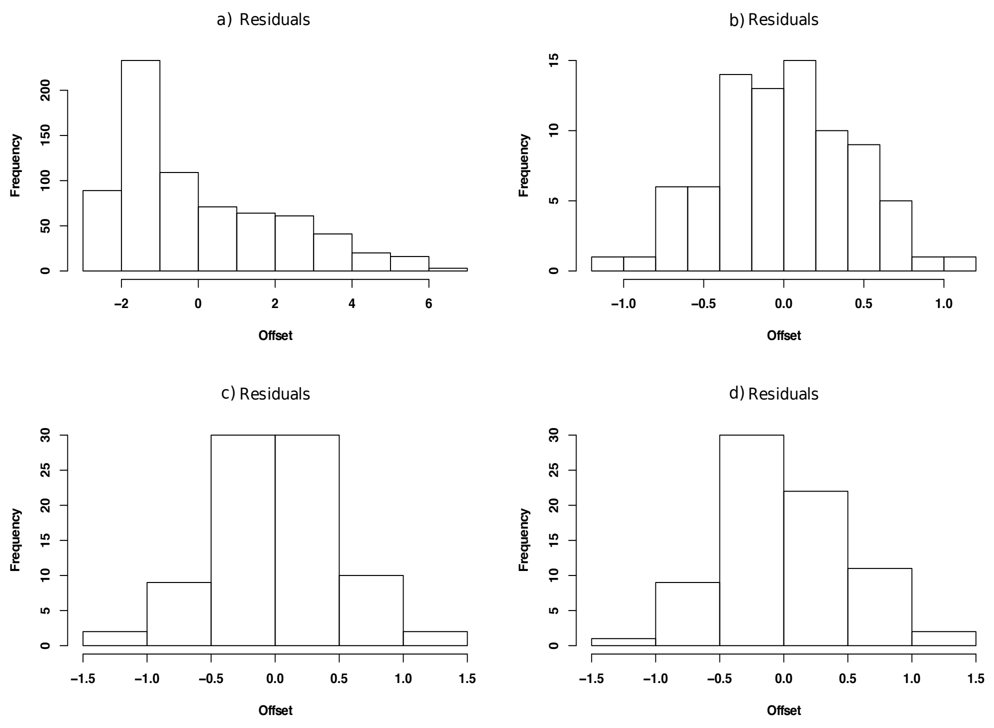


Figure 5.7: Drift measurement for sensor nodes outdoors.

crystal precision (160ppm vs 50ppm). However, the clock offset can be precisely predicted through linear regression. The residuals (Figure 5.6) appear to be normally distributed. The maximum residual error never exceeded 1ms.

Figure 5.5b shows the clock offsets for the experiment conducted outdoors with a different ambient temperature. The clock skew was slightly larger (208ppm). Some values appear lower than indoors, because the low temperature compensated faster crystals. The clock offset became less predictable (Fig 5.7) with residual errors as high as 6ms for one of the nodes. The rest of the nodes did change their frequency, but the residual errors stayed at the same level. As seen from the experiment, the maximum residual error was 6ms. This values was used as duration of wake-up tone in the protocol.

5.5 Schedule Exchange Extension

In this section we evaluate the performance of schedule exchange extension. We compare the performance of a protocol with and without schedule exchange extension for unicast transmission scenario. All the nodes were arranged in a star topology with a receiver in the centre (Figure 5.1). The nodes were booted at random times within 10 seconds. The nodes sent packets every minute and the experiment duration was set to 20 mins. We expect the protocol with schedule

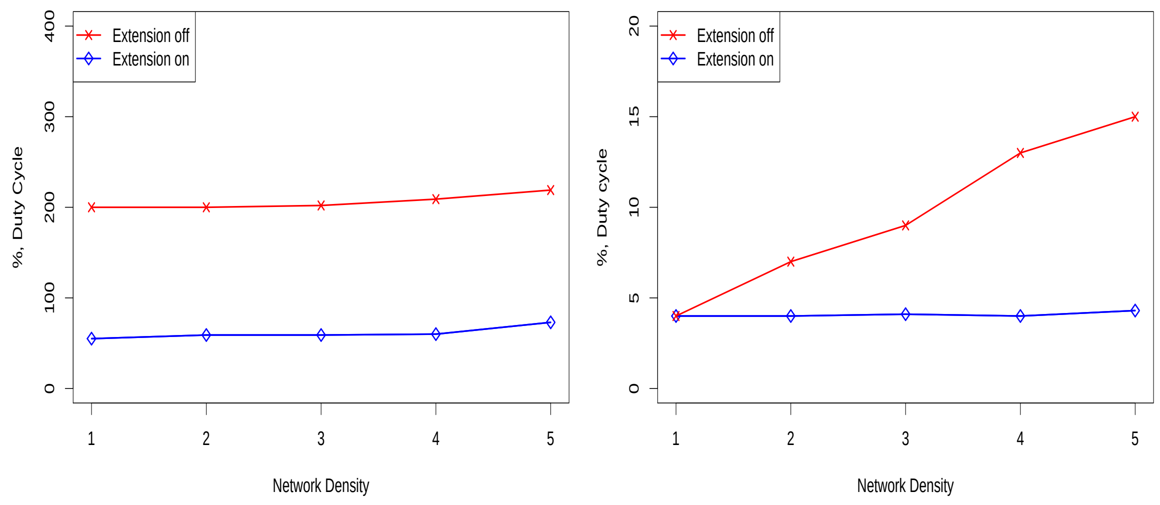


Figure 5.8: Evaluation of Schedule Exchange Extension.

exchange extension to have lower transmission costs due to skew compensation feature and fewer number of broadcasts.

The result of experiment is shown on Figure 5.8a. Figure 5.8b shows the benefit of schedule exchange extension. In the experiment, a node needed to send packets to a randomly chosen node. In a protocol version without extension, each node needs to advertise its presence before the packet exchange can begin. Advertising presence is much more expensive than sending a message itself. With MAC*ron* (with extension), only one random node is required to advertise its presence. Upon receiving a beacon, all nodes respond within a certain time window (60s), after which the beaconing node pushes the list of neighbours to all nodes. This enables all nodes to communicate directly without advertising their presence.

5.6 Summary

We have done extensive evaluation of MAC*ron* protocol through implementation on real sensor nodes, and measurements on a testbed. We have implemented MAC*ron* in TinyOS for Tmote Sky sensor nodes. We have shown through experiments the performance improvements over recent LPL-based techniques such as X-MAC. For unicast transmissions, the power consumption of MAC*ron* is up to 4 times lower than in X-MAC protocol. The power consumption does not depend on network density for broadcast applications, as opposed to X-MAC. We have evaluated a schedule exchange extension option and presented a scenario where it is more efficient.

We have presented the results of local clock drift measurements both indoors and outdoors and shown that the drift can be precisely approximated using recursive linear regression as indicated by the normally distributed residuals (Figure 5.6, 5.7).

The next chapter will describe the evaluation of macro-level and more application oriented duty cycling techniques, which allow the optimization of the power usage through the powering off the devices for large periods of time when observation is not needed (e.g., for some animals or individuals it might be during the night): this would allow additional power conservation.

Chapter 6

Evaluation of Macro-level Scheduling

In the previous chapter we evaluated the performance of a micro-level approach, which introduced a novel time-skew correction and schedule exchange features. In this chapter we present the evaluation of macro-level scheduling approach presented in Chapter 4. The goal of evaluation is to maintain network connectivity at less energy consumption. In order to evaluate this, we conducted a number of experiments in Tossim simulator with several real and synthetic data sets. We have also evaluated the implementation of the protocol reported in Chapter 4 through small scale experiments.

The Chapter is structured as follows. In the first two sections (6.1, 6.2) we evaluate the approach for partially-mobile wireless networks presented in Section 4.5. In the next sections (6.3, 6.4) we evaluate the approach for fully mobile wireless networks presented in Section 4.7. In each section we discuss the simulation settings and parameters used in evaluation. We describe evaluation metrics and the datasets that we used for evaluation and reasons for choosing them. Finally, Section 6.5 presents a critical summary of the evaluation.

6.1 Evaluation for Node Discovery Protocol for Partially Mobile Wireless Networks

In this section we present the evaluation of the protocol for partially mobile wireless networks. To evaluate our approach we used mobility traces to simulate movement of entities around our sinks and evaluated how well the sinks were detecting the movement patterns, by optimizing on their awake times.

The sinks were required to detect mobile tags whenever they got within the communication range of the sinks. We measured the performance of a balanced algorithm and compared it with the performance of two alternative strategies, such as ε-greedy, Boltzmann exploration strategy. The Random strategy, when the sink is spreading its budget evenly throughout the day is used as a baseline.

6.1.1 Settings and Parameters

The evaluation was done by using the R [R Development Core Team, 2007] statistical package. The package facilitates dealing with massive amounts of data, provides a number of statistical functions and is available for free.

The duration of a timeslot was set to one hour. Shorter time slots allow more fine-grained decisions but would have very high variance within each timeslot. Longer time slots have low variance within a timeslot, but will lead to a more coarse-grained decisions. In the extreme case a system would have two time slots (day/night modes). The hourly distributions are stored in a vector $r[1..N_{slots}]$. Each value is updated using EWMA filter as: $r_n = r_{measured} * \alpha + r_{n-1} * (1 - \alpha)$, with α chosen empirically as 0.75. The values are updated only for the active timeslot.

The sinks were given a daily budget of 600s, and needed to distribute their energy according to the daily connectivity patterns. A more detailed sensitivity of the algorithm to various budget settings will be presented in Section 6.2.3 and Section 6.4.

6.1.2 Metrics

We used the number of successfully detected encounters per scanning request as a primary metric for evaluation. For partially mobile wireless networks, where tags only communicate with the sink, this metric can be directly translated into message delivery rate for message dissemination applications.

6.1.3 Dataset

Dartmouth Traces

We used the 802.11 mobility traces from Dartmouth campus [Kotz and Essien, 2005]. As many other campuses in the US, it was equipped with multiple 802.11 wireless access points to provide wireless connectivity to notebook, PDA and mobile phone users. The authentication of users was centralized and every 802.11 association/disassociation event was logged in the traces. As these events reflect the physical proximity of a user to a base station, the traces represent mobility of users within a campus.

For evaluation purposes we selected the busiest access points in the campus and analyzed the traces for a one year period. The trace from a residential building contained 344187 association entries and 307 unique clients (mobile nodes). The trace from an academic building contained 12012 association entries and 527 unique clients.

We made a number of assumptions while using the traces. We assumed that an access point served as a sink, which needed to detect mobile devices, that the mobile nodes are energy constrained and perform periodic carrier sampling. We also assume that the access point is power constrained and needs to save power while being able to detect mobile nodes. In the actual dataset, an energy was not an issue and the access point detected mobile nodes without having to be duty cycled.

Student Traces

We have evaluated our approach on student RFID traces collected during the early tests of RFID tracking system in the Computer Science Department, UCL. During the deployment, 4 people were asked to carry active RFID tags in their pockets or wallets for a duration of 5 days. As a result of an experiment we recorded high resolution raw data about the presence of students in the department, which we used for evaluation.

Badger Traces

We have also evaluated our approach on badger traces collected during the system deployment in Wytham Woods (Oxford, UK). In the deployment 29 RFID readers were deployed in the woods and tracked the location of 37 tagged badgers for several weeks. The RFID readers have been powered on continuously [1] and detected the presence of tagged badgers in the communication range. More details about deployment will be presented in Chapter 7.

6.2 Simulation Results

6.2.1 Impact on Discovery Rate

In the first experiment we compared all four strategies for an access point (AP) located in residential building (Figure 6.1). The strategies were described in Chapter 4 and include random, ϵ-greedy, Boltzmann and Balanced strategies. All adaptive strategies (Figure 6.2) were able to adapt to changing mobility patterns and showed significantly higher performance over simple random one. The ε-greedy strategy performance turned out to be sensitive to exploration rates. Low exploration rates showed better performance; the node learned to select the good time slots and invested some resources into exploring other options. As the node spent more resources into exploration, the performance was decreasing approaching that of a random strategy, as expected. The balanced strategy showed the best performance due to dynamic trade-offs between exploration and exploitation phases.

To evaluate how well all strategies adapt to changing arrival patterns we experimented with the busiest AP located in an academic building. It has different and more dynamic arrival patterns throughout the year depending a lot on the season (Figure 6.3). As we see on the histogram, the distribution of encounters changes rapidly in the 3^{rd} quarter and then returns to normal in the 4^{th} quarter. The sudden change could possibly be related to examination period,

[1]Except when the battery was changed

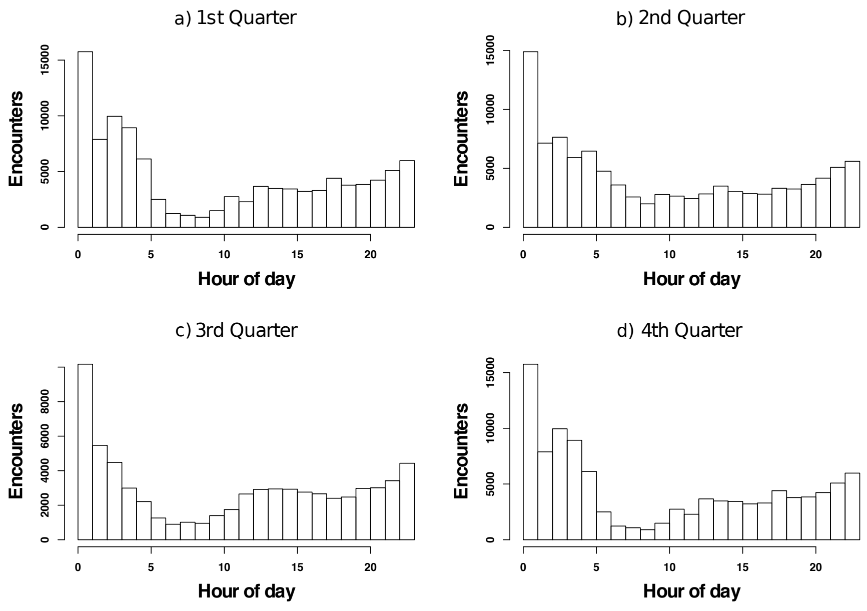


Figure 6.1: Seasonal arrival patterns in the residential building. The activity level peaks at midnight and slows down towards the morning hours. The arrival patterns are relatively similar throughout a year.

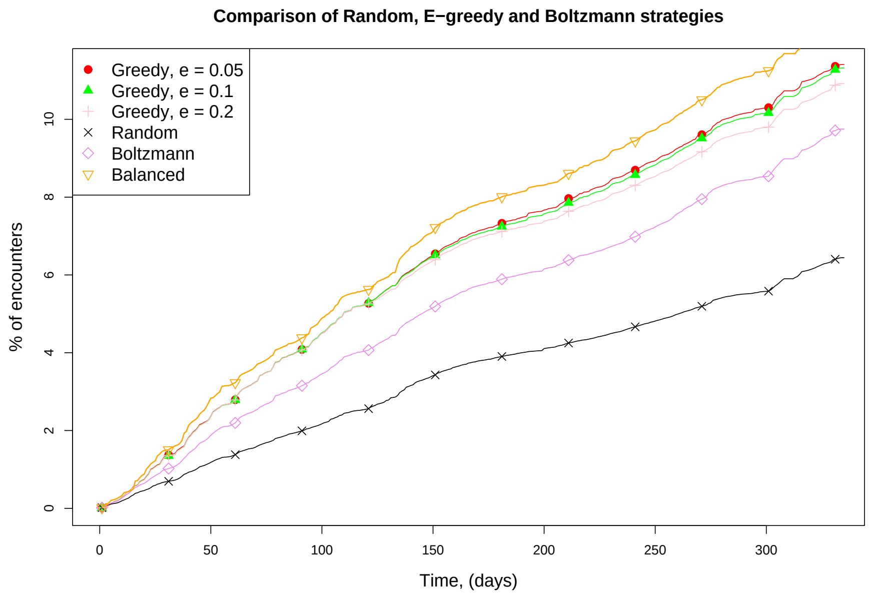


Figure 6.2: Comparison of various wake-up strategies.

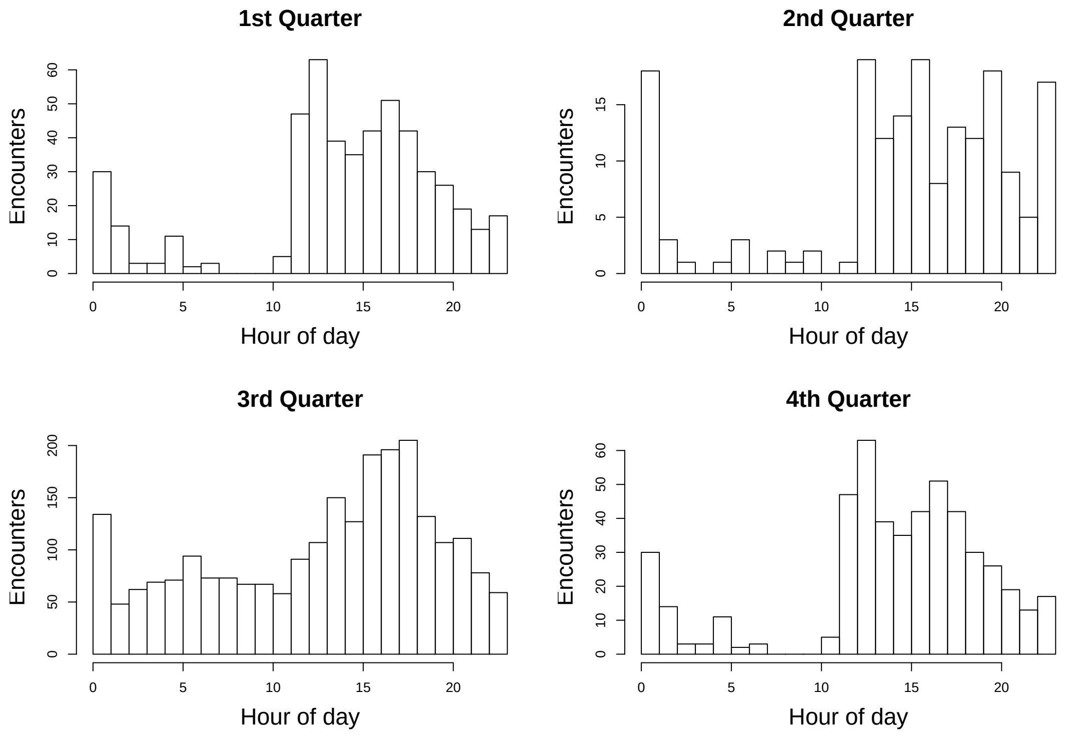


Figure 6.3: Seasonal arrival patterns in an academic building. The arrival pattern changes depending on the season.

when students spend more nights in the labs.

As we see from Figure 6.4, the adaptive strategies performed differently in this set of traces. As opposed to previous experiment, the ε-greedy strategy performed worse than random. We see that at high exploration rates the system adapted well to changes and showed better results, while at low values, the system was very slow to adapt to seasonal changes and sometime performed worse than random. The rapid increase in performance of all four in the second half of the experiment can be explained by much higher load during that time, leading all four strategies to detect more encounters. The balanced strategy managed to maintain a good performance. It adapted quite well to changing traffic patterns.

To summarize, the balanced strategy showed the best overall result for both scenarios. It allocated energy dynamically depending on the expected number of arrivals and adapted well to changing arrival patterns. The ε-greedy performance was very sensitive to exploration rates. The optimal value of exploration rate depends on how dynamic system is, which can be difficult to know in advance. Random and Boltzman performed robustly in all scenarios.

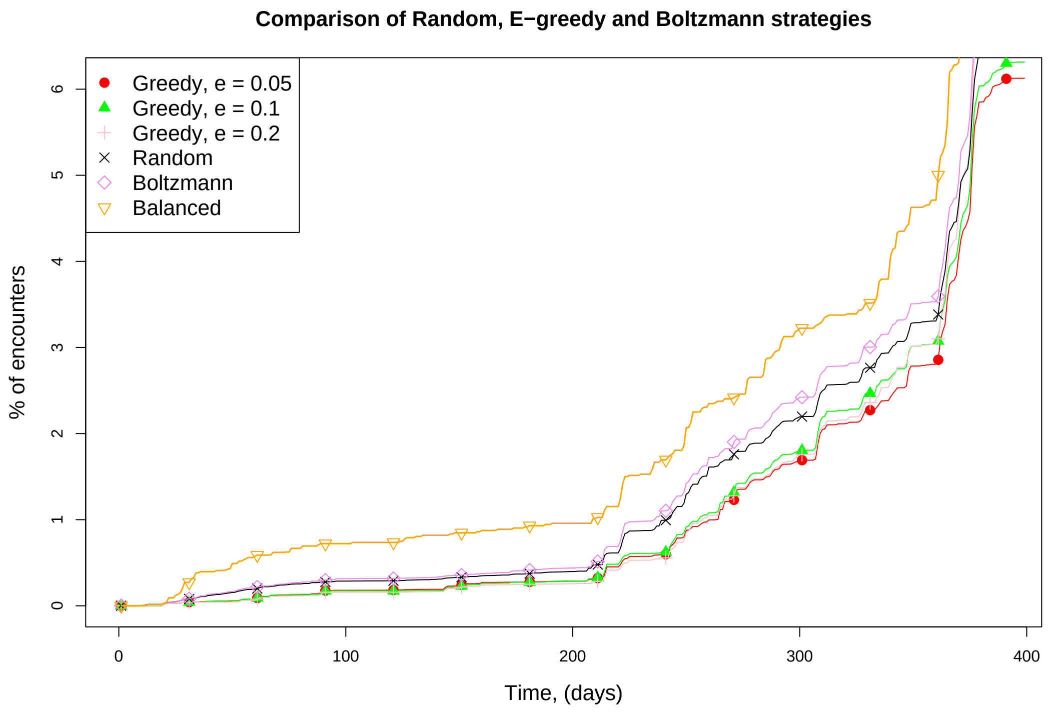


Figure 6.4: Comparison of various wake-up strategies.

6.2.2 Evaluation on students

In this section we present the evaluation of a macro-level object tracking approach on a small group of people carrying active RFID tags. The experiment was conducted in February 2008 in the Computer Science Department, University College London. In the experiment, 4 students were asked to carry active RFID tags in their pockets or wallets for a duration of 5 days. The active tags had default configuration, where they sent beacons periodically every 2.5s with a small random offset. The RFID reader was powered from batteries and worked continuosly at 100% duty cycle. The reader was connected to the sensor node over RS232 interface, which logged all the data into local storage. The data was later retrieved over a wireless link. The original goal of an experiment was testing the newly assembled system for bugs before testing on sheep. However, it also resulted in some useful data, which we used for this evaluation.

As a result of an experiment we recorded high resolution raw data about the presence of students in the department. Figure 6.6 shows the daily activity distribution of people who participated in the experiment. As expected there have been no encounters at night or early in the morning. Most tags had a distinct daily pattern, where they would usually appear in the office at 10am or 12am and stay until 8pm or 10pm. Figure 6.6e) shows the overall pattern,

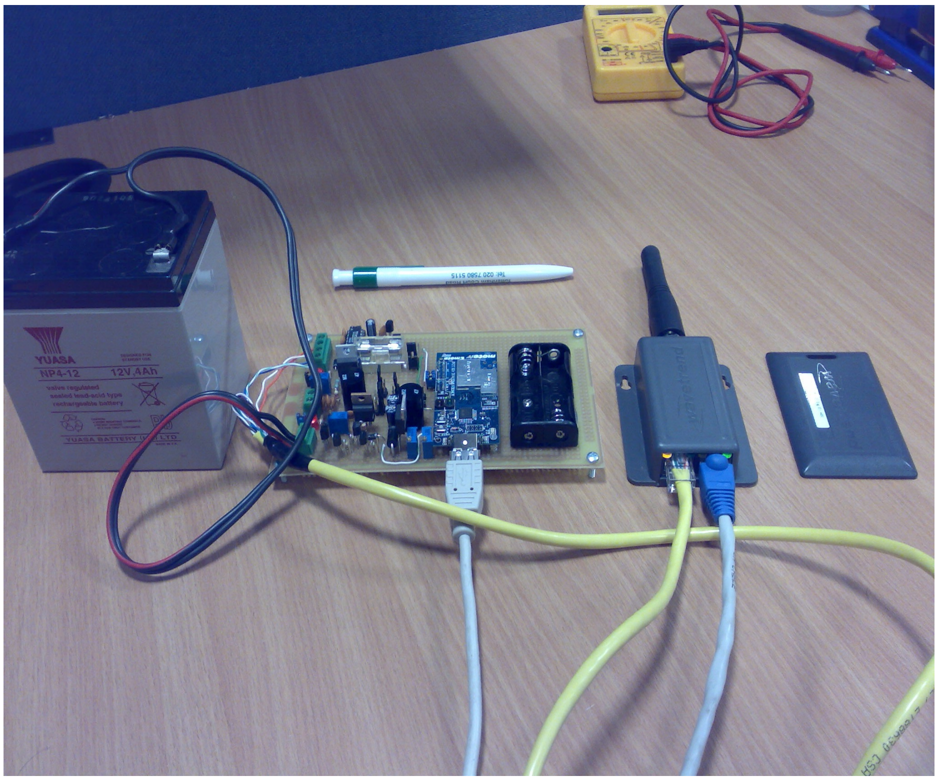


Figure 6.5: A picture of RFID node used during mini-deployment. The deployment was done using personnel tags (right) distributed to several research students in the CS department.

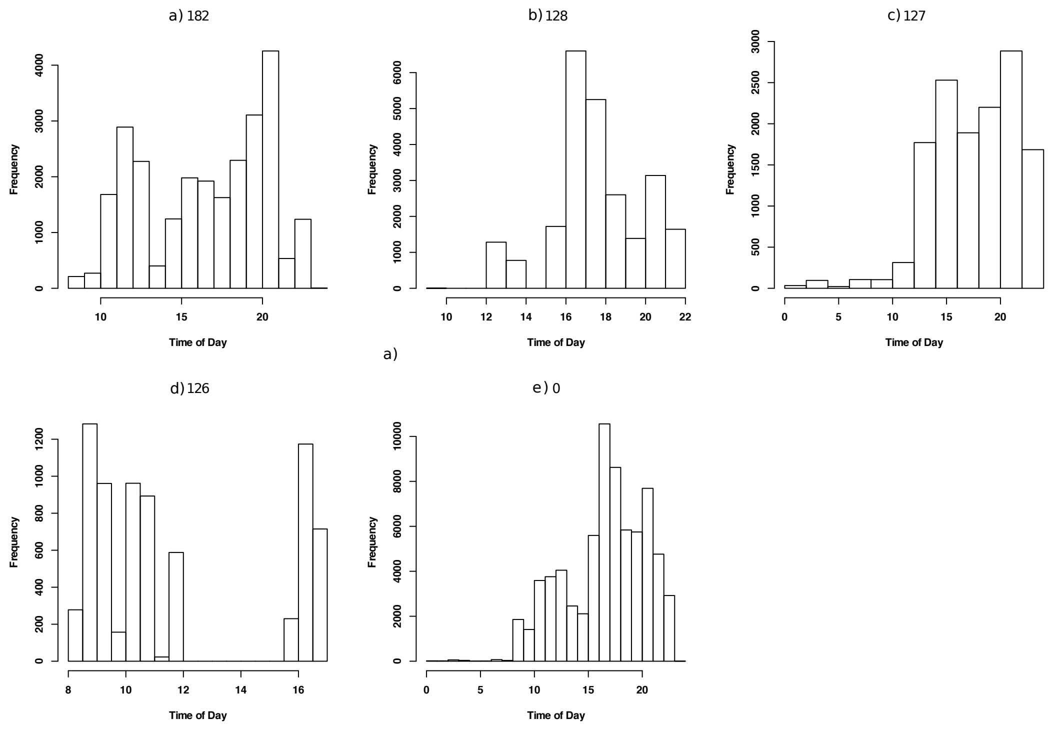


Figure 6.6: Activity histogram of individual tags.

summarizing the data from all tags. It shows that the tags were most active from 3pm to 9pm, with a peak at 4pm. Tag “126” was less active and had a slightly different pattern appearing mostly in the mornings and afternoons.

The original traces contained 2204 encounters with a median contact duration of 7s. This was because most tags were located on the other side of the building with a poor reception. The contacts with such a short duration are very unlikely to be detected by sparse sampling and would have required ’always on’ reader. For the purposes of evaluation, we set a contact timeout to 300s, and the traces resulted in 112 encounters with a median duration of 692s.

These traces were plugged in Tossim simulator to compare the balanced and random protocols. The base station was logging the presence of mobile tags and the scanning period was set to 600s, which is similar to a value used with other traces. The experiment was done to confirm the applicability of an algorithm to human RFID traces. A more complete sensitivity analysis of an algorithm to various budget settings was done on MIT, Intel and badger traces. The experiment was repeated 10 times. The results of comparison are available in Table 6.1. With random scanning, the node was able to detect all 112 encounters, which took 6233 scanning requests in total, which converts to 56 beacon requests per detected encounter. The balanced detected 89 encounters consuming 4157 of scanning requests, which is equivalent to 46 beacon requests per detected encounter. In other words, the balanced scanning detected fewer encounters, but also consumed fewer scanning requests with an overall optimisation of 16%.

6.2.3 Evaluation on badgers

Figure 6.7 shows the performance comparison of fixed and balanced duty cycling for various budgets. We experimented with budgets ranging from 9% to 50% by modifying the sleep time of the reader. Figure 6.7a shows the total number of encounters detected by both algorithms for various budgets. It shows that the fixed algorithm detected more encounters on average, than the balanced strategy. Figure 6.7b shows that the *actual* duty cycling of an balanced algorithm was significantly lower than that of a fixed node. In other words, the balanced algorithm suppresses scanning during the day resulting in the actual duty cycle lower than the programmed budget. Finally, Figure 6.7c shows the number of encounters normalized by the duty cycle, which is

Table 6.1: Results of macro-level evaluation on student RFID traces

Protocol	**Encounters**	**Collocation, hours**	**Beacon requests**	**Beacons/Encounters**	**Beacons/Collocation**
Random	112	45	6233	56	138
Balanced	89	41	4157	47	101

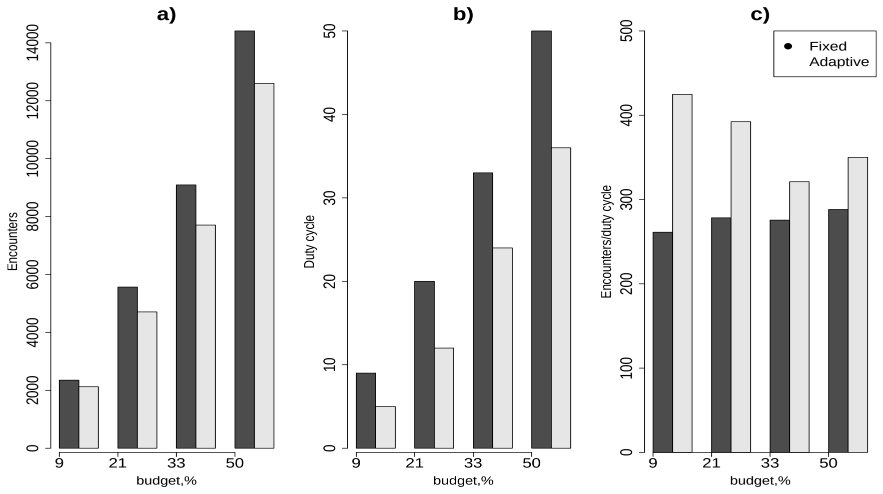


Figure 6.7: a) the number of encounters for different budgets. b) actual energy consumed c) the number of encounters per duty cycle. The balanced detects similar number of encounters but has a higher number of encounters per effective duty cycle. [Data from 1st deployment]

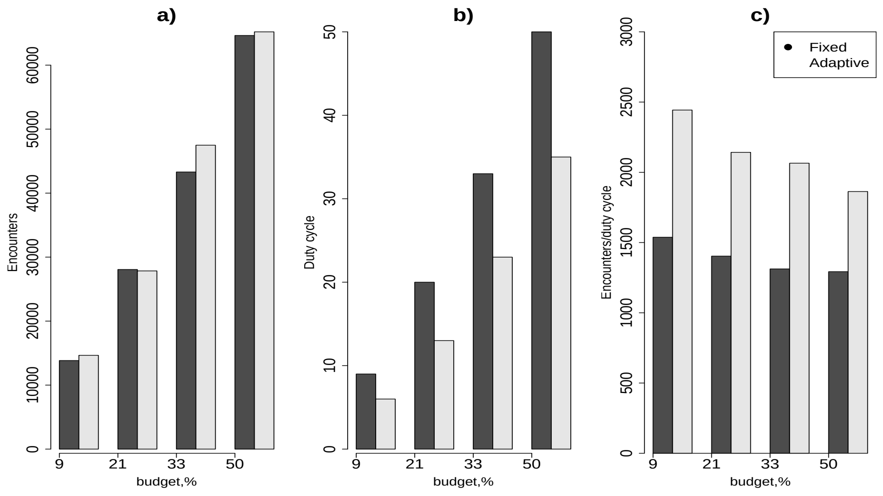


Figure 6.8: a) the number of encounters for different budgets. b) actual energy consumed c) the number of encounters per duty cycle. The balanced detects similar number of encounters but has a higher number of encounters per effective duty cycle. [Data from 2nd deployment]

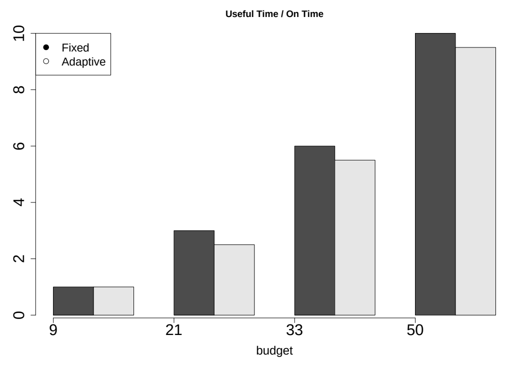


Figure 6.9: The proportion of useful times by different algorithm.

equivalent to number of encounters per unit of energy. It clearly demonstrates that balanced algorithm detects more encounters than the simple fixed strategy in the entire range of budget values used in the experiment. Figure 6.8 shows the results from a second deployment, which are consistent with previous result.

Figure 6.9 shows the amount of useful time spent by different protocols for a range of budget settings. Even though the balanced strategy worked at a lower duty cycle, it managed to spend almost equal amount of time detecting badgers.

6.3 Evaluation for Fully Mobile Extension

In this section we evaluate the performance of macro-level approach for fully mobile wireless networks presented in Section 4.7. The goal of performance evaluation is to measure the impact of macro-level adaptation on network connectivity and network performance.

This section is structured as follows. We first describe simulation settings and parameters including metrics and the description of datasets (Subsection 6.3). We then describe the results of performance evaluation (Subsection 6.4). We conclude by summarizing the results and doing a critical evaluation.

6.3.1 Settings and Parameters

Tossim is a discrete event network simulator used to simulate TinyOS applications. The simulator provides own implementations of hardware components at different levels of granularity. The timers are simulated at hardware level, while the radio is currently simulated at a packet level. It is implemented as library and requires just a recompilation of an application for the simulator platform. The simulation is scripted from a Python or C++ program, which sets up simulation parameters, starts the simulation and generates network connectivity events.

Running simulations over several days or months of traces such as Dartmouth traces described previously, required changing the default parameters and settings. The external program running the simulation was written in C++ as opposed to Python, the graphical interface was disabled and all the data messages were generated by the sensors instead of injecting from the simulator. As a discrete event simulator, the simulation speed depends mostly on application complexity in terms of event rate the application generates. All events are generated either by a timer or I/O interrupt, such as sending or receiving a packet. Both applications we evaluated are relatively lightweight in this respect. Since a sparse node is isolated most of the time, the node would usually generate few events when sending or responding to beacons.

Tossim does not have direct support for mobility, but allows dynamically adding, removing and changing communication links between the nodes from a C++ program dynamically during the simulation. The actual mobility can be provided by an external program, such as a Tython script, which translates node coordinates into communication links as the nodes change their position. However, this translation process takes a very long time, as communication link has to be re-evaluated for each node pair in the network at every timestep. To speed up the simulations, the traces were preformatted into a stream of connect / disconnect events, so that no distance recalculation was required while the simulation was running. In the simulation, the nodes used standard throughput of 250kbps and default CSMA MAC protocol.

Since the simulation was done with real datasets, the simulation settings such as communication range and node speed were inherently defined by the datasets and methods used to collect the traces, which will be discussed later in Subsection 6.3.3.

In the evaluation, we focus on the impact of adaptive scanning on the discovery of new nodes and the performance of network applications. The node discovery process actively

Table 6.2: Simulation parameters

Parameter	**Value**
Packet size	28 bytes
Data rate	250 kbps
MAC	CSMA
Maximum number of nodes	8
Communication range	defined by traces
Nodes	defined by traces
Traffic load	once an hour sent to a random sink
Timeslot size	1-12 hours

changes the network topology. The higher the sampling frequency the more encounters the nodes will be able to detect, therefore giving nodes more opportunities to communicate. It is therefore natural that we select message dissemination and encounter tracking application to evaluate the performance of the macro-level approach. Both of these applications have been implemented for TinyOS and were run in TOSSIM network simulator.

6.3.1.1 Message Delivery Application

In the message delivery application the nodes were configured to generate one message per hour and send it towards one of several sinks using a direct delivery algorithm, in which a sender delivers a message directly upon an encounter with a destination node (e.g., a sink in our scenario). Upon successful delivery the message is acknowledged by the sink and removed from the sender's buffer.

The nodes serving as sinks were chosen randomly at each simulation run. The message was considered delivered when it reached at least one of the sinks. If the message was delivered to several sinks, it still counted as one delivery. For the purpose of evaluation of adaptive duty cycling we set a buffer size large enough to avoid message dropping and thus excluding its impact on the performance of the message delivery application. In practice, the buffer size can be unlimited due to availability of inexpensive flash and SD card memory chips. The buffer size was set to be 1000 messages. The queuing discipline was set to FIFO, the messages were not deleted from the buffer unless they were delivered and then acknowledged by one of the sinks. The nodes start generating data messages immediately after the start of the simulation and keep the messages in the buffer until delivered.

6.3.1.2 Encounter Tracking Application

The encounter tracking application registers and records all encounters between individual nodes. Figure 6.10 outlines the discovery process. The line on top displays the physical presence of nodes A, B, C and D in time. The line in the middle displays the sampling process by node E. Thick bars represent discovery beacons, while thinner bars depict keep-alive messages. The line on the bottom represents the node presence, as preceived by node E. Without a notion of a continuous encounter, it is not possible to compare the performance of various discovery methods. In other words, more frequent beaconing might 'discover' the same physical encounter several times, which would not be correct because it would be the same physical encounter.

Therefore, it is important to differentiate between *encounter* and *discovery*. An *encounter* is a continuous period of time, when the node is in communication range. *Discovery* is an event

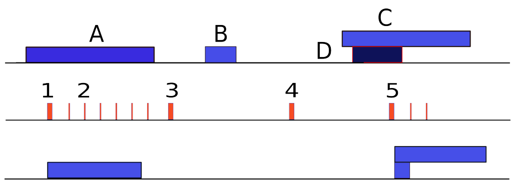


Figure 6.10: Encounter tracking diagram: 1 - node A is discovered. 2 - node E keeps track of A by using frequent and less expensive keepalive messages. 3 - node A goes out of range. 4 - no nodes are around; encounter with node B is missed. 5 - nodes C and D are discovered.

by which the encounters are detected. We experimented with beacon periods from 600[2] to 79200 seconds. It should be noted that for long beacon periods (e.g., 1 hour), if a node receives 3 consecutive beacons, there is no way of telling whether that is one continuous encounter lasting 3 hours or just three shorter encounters, thus making evaluation difficult. For this reason, we also introduce short keep-alive messages in the application protocol: once the nodes discover each other they remain synchronized and exchange keep-alive messages every 60s. This makes it possible to track encounters with more precision. If a node does not hear from other node for a certain amount of time, called *expiry time*, it considers an encounter to be finished. The encounter expiry time was set to 300s.

For each encounter the application logs an encounter time, duration and the neighbouring node. The information about encounters is logged using TOSSIM debug interface as text messages and then processed using a separate script.

6.3.2 Metrics

We are using the beacon scanning rate as an estimate of an energy consumption. First, scanning is an energy expensive process because the nodes need to continuously stay awake in either listening or transmitting state. Second, scanning is represented by a separate state with known power consumption. For example, the cost of operating in INQUIRY mode is twice as expensive as in normal mode. Similarly, the extended preamble takes more power than operating in sleep mode for 802.15.4 networks. Therefore, the power consumption is directly dependent on the beacon scanning rate.

For evaluation purposes, we consider low data rate applications. Typical environment monitoring applications fall into this category. For example, monitoring a temperature and humidity at 5 minute intervals generates 1152 bytes per day assuming each reading requires 2 bytes. In terms of transmission time, this amount of information requires 36ms. The usage of compression techniques could reduce this number even further. For comparison, let us consider

[2]This is a sampling rate of original Intel traces.

the amount of traffic generated by a discovery process. This depends on scanning rate, the duration of an extended preamble and data rate of the radio. If a node scans once per hour with a 1s long preamble, that is equivalent to 4Mb of traffic per day, which is substantially higher than the amount of traffic generated by applications that we are considering. In terms of time, this is equivalent to 24s of continuous transmission.

We are interested in how the adaptive scanning affects the performance of distributed application. Therefore the metrics should reflect how well the application is performing as a function of scanning rate. Hence, we use the following metrics:

- **Message deliveries per scan rate**. We expect a balanced scanning algorithm should provide the same message delivery rates with fewer scanning requests.

- **Encounters per scan rate**. We expect a balanced scanning algorithm should detect the same number of encounters by using fewer scanning requests.

Thus, we ignore the effects related to radio propagation, interference and packet loss. Packet loss can occur due to collisions, interference and noise. Packet loss can lead to misinterpretation of a scanning request as noise by the receiving node, which is not critical for detection. The loss of an ack is more critical and can prevent a discovery. This is seen as a general problem with using extended preamble for discovery process and has been discussed in [Chen et al., 2001]. The noise can increase the probability of false positives in the carrier sampling process, which might potentially increase the energy consumption. Upon detecting an energy in the channel, a node needs to spend energy to wait till the end of transmission and reply with an ack. Minimizing the false positive rate has been discussed in [Polastre et al., 2004] and include simple techniques, such as considering only minimum sample values. The intuition behind that rule is that in the presence of transmission, all the samples will have a high RSSI value.

6.3.3 Datasets

We first tested the protocol on synthetic traces, where we generated random encounters between the nodes in the network. The purpose of using synthetic traces is to test the protocol under known conditions, with known and controllable distribution of encounters, their durations and number of nodes. As opposed to real traces, synthetic traces can be generated in large quantities, with a variable parameters, such as encounter rates, patterns, etc.

We then tested the protocol on real mobility traces. The human proximity traces are available and we used the data set from Intel [Scott et al., 2006] and MIT Reality Mining projects [Eagle and Pentland, 2006]. The reasons behind selecting those data sets is their availability and relatively large number of nodes in the experiment. Both projects used Bluetooth devices to log the encounters between individual people. The traces were used to model the presence of communication channel between individual sensors. Due to power limitations, the original human mobility traces were results of sampling every 600s and 300s respectively, which might have missed some encounters and introduced a certain granularity of encounter duration. In this thesis, however, we assume that the traces represent ground truth data about physical movement of entities and that our optimal result would be to detect all contacts. Finally, we tested an approach on mobility traces of wildlife animals, brushtail possums, native of Australia and Tasmania [Dennis et al., 2008]. Below we give more details about each dataset.

6.3.3.1 Random Graph Dataset

The advantage of synthetic traces is that they can be generated with certain known properties, such as network density, temporal patterns and dynamicity to test and evaluate the protocol over a wide range of parameters, which is not possible to do with real traces due to their limited availability. The random graph was generated as follows. At each timestep, the connection between any random nodes was established for a duration between 300 and 900s, which was uniformly distributed. We generated the traces for 36 nodes over a duration of 90 days.

To model dynamic environment the network operated according to one of two schedules. In schedule A, all the encounters were established between 8am and 3pm; in schedule B, all the encounters were established between 22pm and 5am. We then generated 5 sets of traces, where schedules were alternated every 15, 30, 45, 60 and 75 days, which represented various levels of temporal dynamicity in the network. To model various network densities we generated another X sets of traces, where average number of encounters per node varied from 2 to 16 encounters per day.

6.3.3.2 Intel Dataset

In Intel dataset, the traces were collected using a group of 36 students around in Cambridge, UK carrying small Bluetooth devices (Intel iMotes) over a period of 11 days. The purpose of an experiment was human mobility characterization for content distribution evaluation in Delay Tolerant Networks. The sensor nodes were programmed to record the opportunistic sightings of any other Bluetooth devices in the communication range (typically within 10m) and log this information into local flash memory. The scanning time ranged from 5 to 10 seconds with most encounters taking about 5s to discover. The period of scanning requests was randomized to avoid potential synchronization of scanning requests[3]. Due to limited battery capacity (CR-2,

[3]In Bluetooth protocol, a node in active scanning state is not able to respond to other scanning requests.

Figure 6.11: An image of a brushtail possum.

950mAh), the duration of experiment was limited to 11 days.

6.3.3.3 MIT Reality Mining Dataset

The use of mobile phones, which can be recharged daily, enabled collection mobility traces for much longer duration of time. In MIT Reality Mining dataset, the traces were collected using 96 people carrying Bluetooth mobile phones over a duration of 292 days. The dataset represents over 350,000 hours of continuous data on human behaviour. The phones were programmed to register the opportunistic sightings of other phones, cell tower ID, phone status (charging or idle) and application usage. These traces, therefore, reflect the daily activity patterns of humans over extended period of time. The evaluation was done on 60 most active nodes over 3 months of traces. We excluded users, which had very few or did not have any encounters, which was caused by either software bugs or users leaving their phones on their desks.

6.3.3.4 Possum GPS Mobility Traces

While human traces are available relatively in abundance, animal traces are still scarce resouce and the possum GPS traces were among very few traces available for this study.

Brushtail possums (Figure 6.11) are small nocturnal animals native to Australia and Tasmania. In the original study, the animals were tagged [Dennis et al., 2008] with small (105g) GPS loggers (Sirtrack Ltd., New Zealand), configured to record GPS fixes every 15 mins. The data was stored locally on the tag, and retrieved after re-capturing an animal. Each animal had to be recaptured every 2 weeks throughout a year to recharge or replace the batteries. The pos-

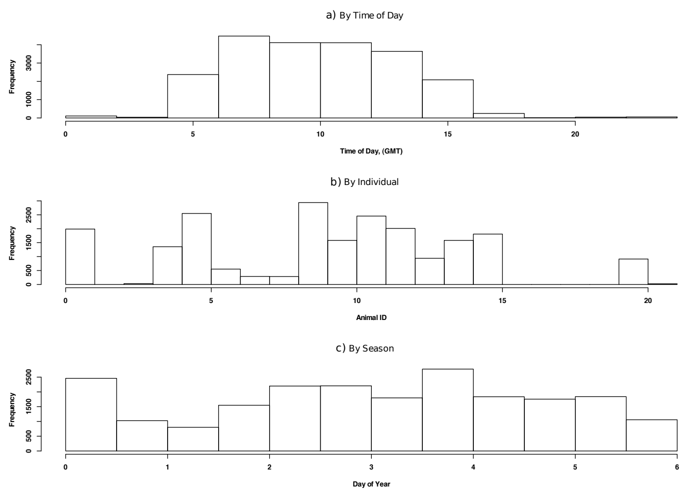


Figure 6.12: Histogram of node activity by time of day, node ID and season. The activity levels show correlation with time of day.

sum is a nocturnal animal and becomes active after a sunset with peak activity between 11pm and 2:30am. During the day, the animal hides in dens, located in tree hollows or cavities. Figure 6.12a shows the probability of a possum encountering another possum depending on time of day. This probability depends on time of day, different individuals and season, which makes it a good candidate for our evaluation. To extend the lifetime of a deployment the tags were scheduled to operate for 11 hours a day, when the possums are active.

The proximity of dense forest and weather conditions affected the GPS performance leading to an overall 87.6% GPS fix rate leading to missing values in the traces. When converting the traces into proximity information we first extrapolated the traces by filling the missing fixes with the last known position. For the purposes of evaluation, we assume the nodes remain stationary during the day in the last recorded position. At each timestep, the spherical distance between every node pair was calculated using Haversine formula: [Sinnott, 1984]

$$d = 2R\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos\phi_1 \cos\phi_2 \sin^2\left(\frac{\Delta\lambda}{2}\right)} \tag{6.1}$$

Where d is the spherical distance between two points, R is the average Earth radius, ϕ_1, λ_1 and ϕ_2, λ_2 are the geographical latitude and longitude of two points.

If the distance was shorter than a certain range, an encounter was registered between the corresponding two nodes. The threshold range was chosen to be 50m, which reflects the capabilities of modern radio and the range was small enough to carry a biological meaning. The evaluation was done over 180 days of the traces with 14 animals for encounter tracking appli-

Table 6.3: Description of traces used in evaluation.

Source	**Nodes**	**Duration, days**	**Sampling interval, s**
MIT Reality Mining	96	292	300
Intel Haggle	36	11	600
Possum traces	14	180	900
Random data set	36	90	600

cation and 40 days for message delivery application.

All datasets have been converted into connect / disconnect format as shown below and plugged into the Tossim simulator.

```
<TIMESTAMP> <ID1> <ID2> <CONNECT/DISCONNECT>
```

The summary information about all the traces including the number of nodes, duration and sampling interval is presented in Table 6.3.

6.4 Simulation Results

6.4.1 Message Delivery Rates

Figure 6.13a compares the message delivery rates for balanced and random strategies for MIT traces. The X axis represented the average scanning period, which varied from 7220 to 72000s, axis Y depicts the number of successful deliveries using direct delivery application in percentage to total number of messages.

Both strategies show very similar performance, the delivery rates drop linearly with increasing scanning interval with balanced slightly outperforming random. At a scanning period of 72000s the nodes are still able to deliver at least 10% of messages. This number is likely to be much higher for multi-hop and replication based protocols.

Figure 6.13b shows the average number of beacon per delivery for the same experiment. It shows that while balanced provides higher delivery rates, it consumed considerably fewer beacons. The ratio drops rapidly and then keeps at a steady level. The low frequency scanning focuses on few very stable timeslots, which results in a plateau. The higher the frequency of scanning, the more frequently the nodes will explore potentially interesting timeslots, thus increasing the beacon to delivery ratio. This explanation is supported by results on random graph traces, where in the absence of a stable component in mobility traces, the beacon per message delivery ratio remains constant throughout all scanning frequencies. It is interesting to note that at a beacon rate of one beacon per day, balanced strategy still maintains a delivery rate of about 40%, more than twice that of random strategy.

The results on Intel traces (6.14b, 6.14a) show similar behaviour and consistent with previous findings. As opposed to MIT traces, the balanced strategy provided considerably higher deliveries than random strategy, which can be explained by presence of stronger patterns in Intel traces.

The results of simulation on Possum traces are presented on Figures 6.15a and 6.15b. The balanced strategy provides more deliveries than random. The efficiency, in terms of beacon per delivery was also higher for short beacon periods, but became almost the same as random for longer scanning periods.

6.4.2 Discovery Rates

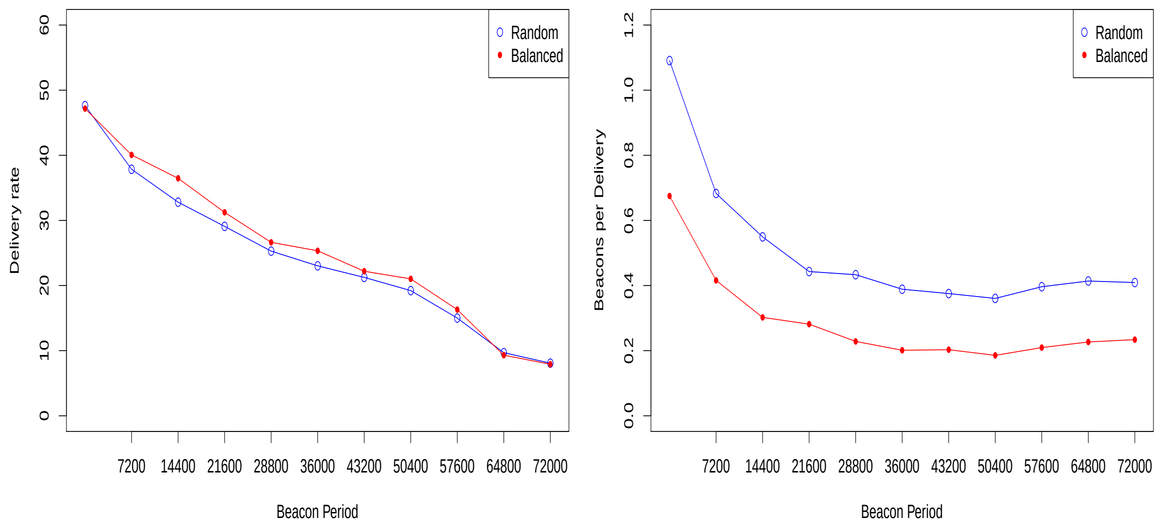
In the following experiment we evaluate the impact of adaptive scanning rates on the performance of encounter logging application.

Figures 6.16a shows the number of detected encounters for scanning intervals from 7200 to 72000 seconds for MIT Reality Mining traces. The graph shows that the balanced strategy detects more encounters than simple random strategy. It shows that while balanced detected more encounters, they consumed much fewer beacons (Figure 6.16b).

In the course of experiments, we observed that the number of detected encounters of the balanced algorithm depends on the maximum and the minimum number of beacons in one timeslot. In the experiment we set it to maximum of 130% and 10% of average (budgeted) scanning rate. All the graphs show the percentage of encounters detected by the 2 algorithms over the total number of encounters in the traces. We then tested the algorithm sensitivity for different timeslot durations and found that longer timeslots perform better for lower scanning rates. In this experiment the nodes used 3 hour timeslots.

Figure 6.18a and 6.18b show the results for the GPS possum traces. The results for the possum traces are consistent with human traces. The approach managed to learn the possum's activity pattern and reduce the scanning rate at a day time, when possums are not active. However, the performance advantage of balanced duty cycling is lower than for human mobility traces. The balanced strategy delivered lower number of messages, but was more efficient. The balanced proved most efficient at higher scanning rates and its performance degrades with lower scanning rates.

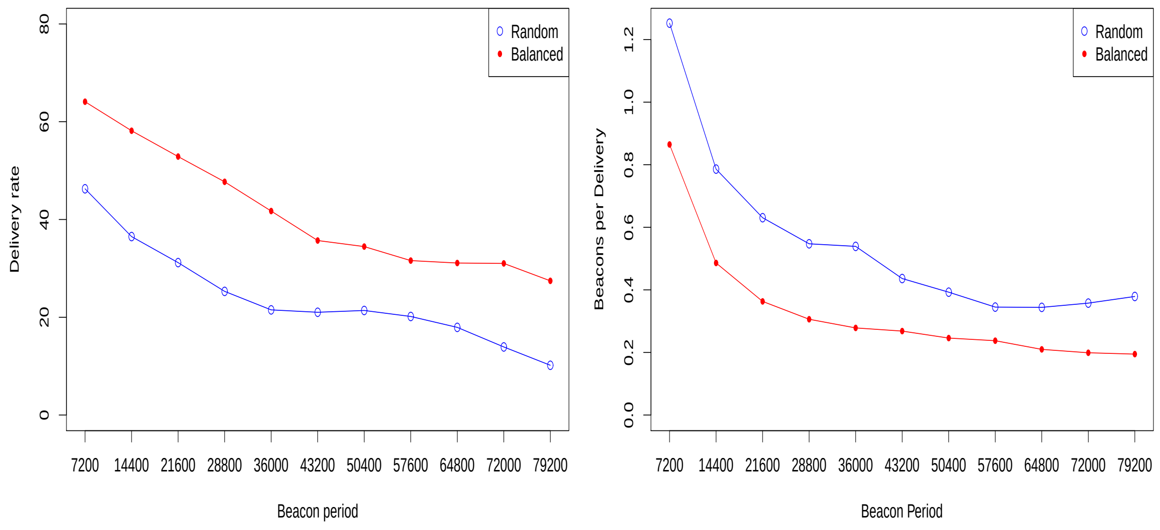
On the other hand, the approach needed to be adapted for each particular trace. In particular, the optimal values of maximum and minimum scanning rates were different for each data set. The minimum scanning rate is needed for less predictable conditions, where it forces the



(a) Deliveries.

(b) Beacons per Delivery.

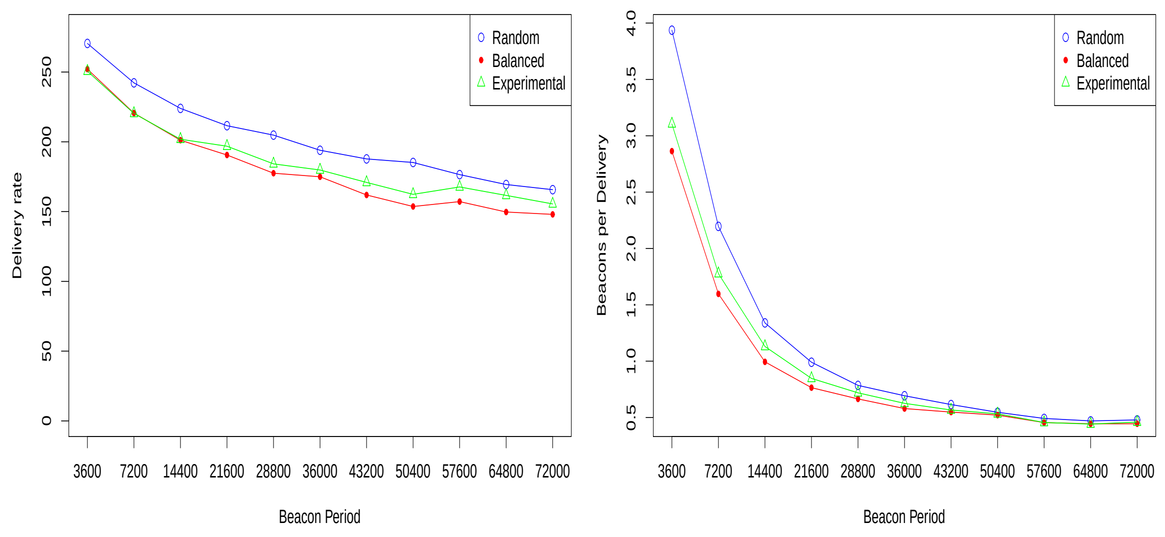
Figure 6.13: Message delivery rates, MIT Reality Mining traces.



(a) Deliveries.

(b) Beacons per Delivery.

Figure 6.14: Message delivery rates, Intel traces.



(a) Encounters.

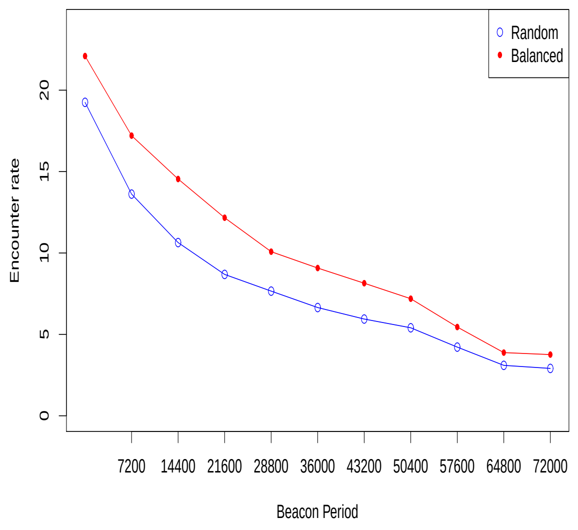
(b) Beacons per Encounter.

Figure 6.15: Message delivery rates, Possum GPS traces.

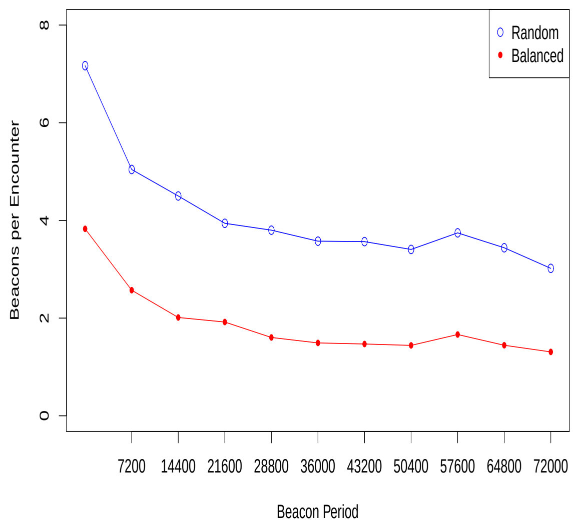
node to keep scanning inactive time slots at a certain minimum rate. The maximum scanning rate is needed to limit the impact of certain timeslots, for example, in a situation, where there is only one active timeslot, with one continuous encounter, it is enough to scan only once, because the subsequent scannings during the same timeslot will detect the same encounter. The maximum scanning rates prevents the balanced strategy from spending all its budget into this timeslot. Automatic configuration of maximumum and minimum scanning rate for each trace is a potential future work.

6.4.3 Evaluation of Experimental Strategy

The results of experimental strategy are presented in Fig 6.15 and Fig 6.18. The preliminary results showed similar behaviour on all data sets and we present the results based on possum traces. It showed that taking into account statistics about individual nodes did not give significant advantage compared to balanced strategy. It provided the same delivery rate as an balanced strategy for higher beacon rates and approached the delivery rate of random for longer beacon periods. The number of beacon requests (energy) per delivered messages is lower than for random but higher than for balanced strategy. The explanation for this behaviour is that the sinks follow the same temporal pattern as the rest of the nodes. Therefore extracting temporal patterns of specific nodes leads to the same allocation of a duty cycle.

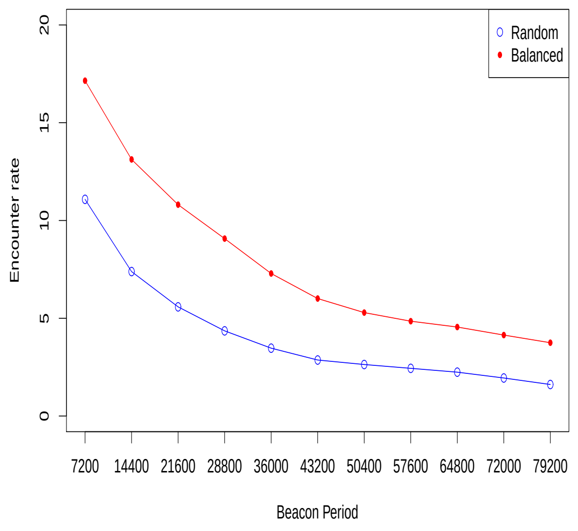


(a) Encounters.

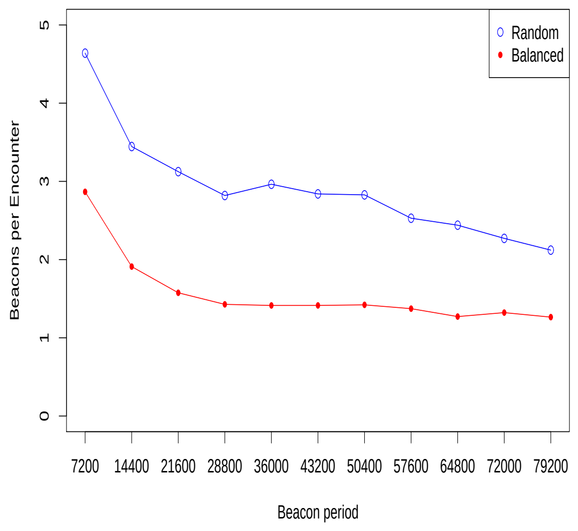


(b) Beacons per Encounter.

Figure 6.16: Node discovery rates, MIT Reality Mining traces.

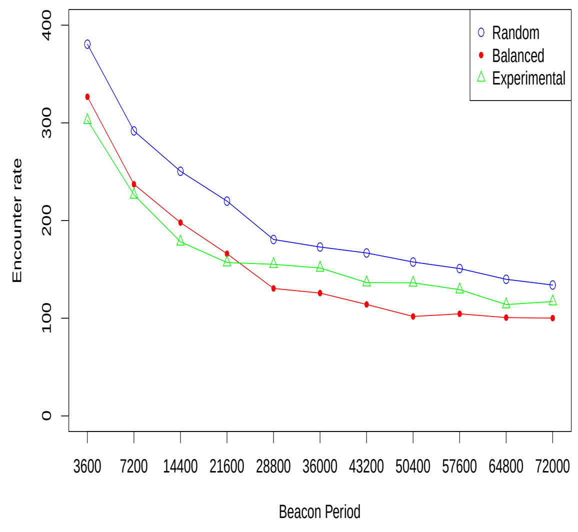


(a) Encounters.

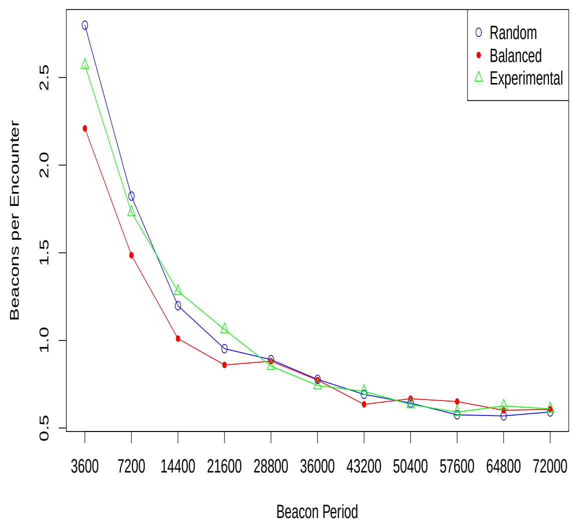


(b) Beacons per Encounter.

Figure 6.17: Node discovery rates, Intel traces.



(a) Encounters.



(b) Beacons per Encounter.

Figure 6.18: Node discovery rates, Possum GPS traces.

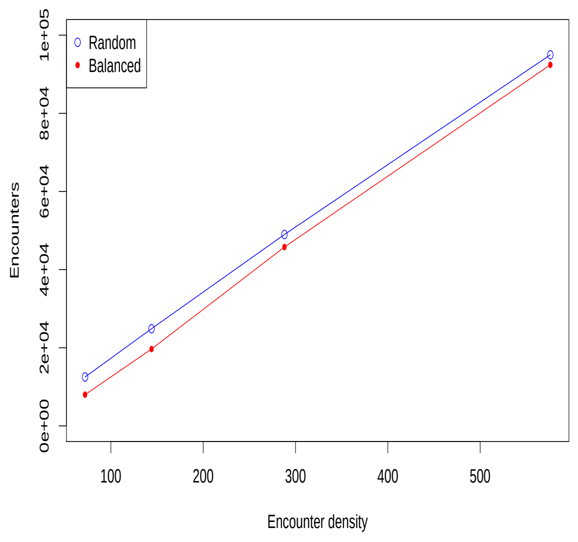
6.4.4 Sensitivity to Network Densities and Network Dynamics

6.4.4.1 Sensitivity to Network Densities

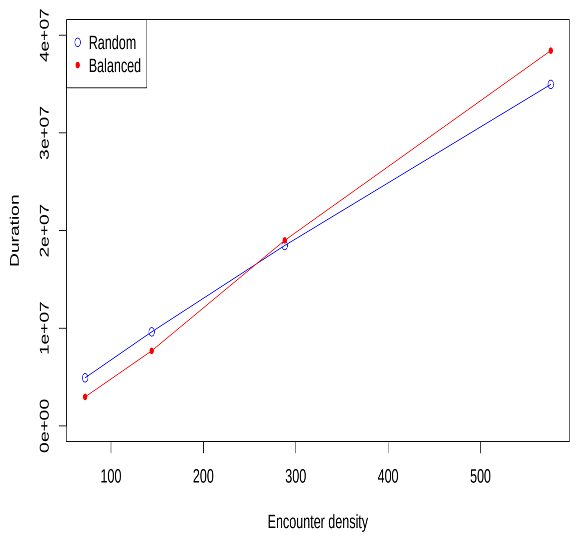
Using synthetic traces allows to validate the protocol behavior on the traces with known and controllable parameters such as network density, or level of dynamicity in the network. In the following experiment we generated a set of synthetic traces, where the network density was varied from 2 to 16 encounters per node per day and its impact was observed on the performance of encounter tracking application. We measure the number of detected encounters and the total duration of collocated time. The average beacon rate was 600s.

Figures 6.19a and 6.19b show the detected encounters and connectivity durations for various network density. There is almost a linear relationship between the number of encounters and the network density. The number of encounters increases because the nodes are more likely to have an encounter with another node. Figures 6.20a and 6.20b show the number of scanning requests per encounter and the number of scanning requests per unit of collocation time, respectively. The graphs demonstrate that the protocol is more efficient than random scanning throughout a wide range of network density.

In sparse scenarios (Fig 6.19a), the balanced strategy maintains lower beacon per encounter, and the total number of encounters can be lower than for random strategy. This happens



(a) Encounters.



(b) Durations.

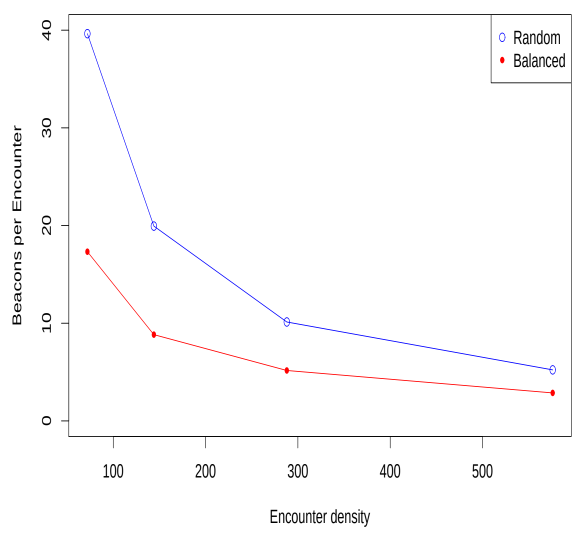
Figure 6.19: Evaluation with random traces. Impact of variable network density.

because the protocol focuses scanning on very few timeslots and suppresses scanning during the rest of the day. In more dense scenarios, the balanced protocol consistently detects more encounters for long durations, while consuming less power.

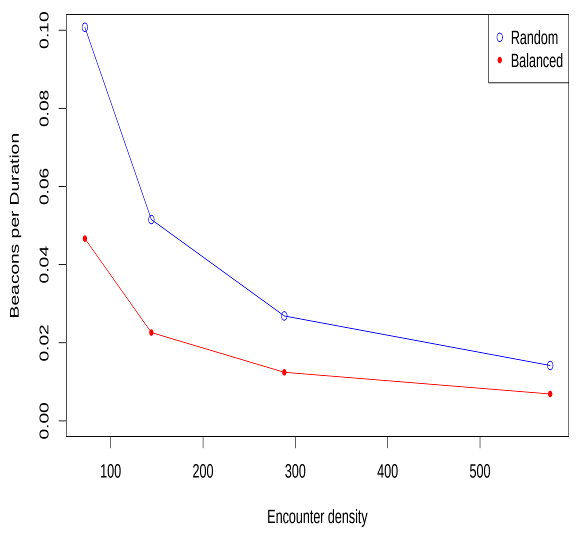
6.4.4.2 Sensitivity to Schedule Rates

In this experiment we evaluate the approach adapts to dynamic networks, with no stable temporal patterns. The performance of the approach will depend on how quickly it converges to a given temporal pattern. We generated a set of traces, where the network schedule was dynamically changed from every 15 to every 75 days and evaluated its impact on encounter tracking application.

Figure 6.21a shows the number of encounters for variable schedule rates. The X axis displays the schedule change rate, and the Y axis displays the number of detected encounters. The average beacon rate was 600s. The number of detected encounters for random strategy remains constant regardless of schedule change rate. This is expected, the random strategy does not adapt to patterns. The balanced strategy shows slightly less performance at higher schedule change rates and then becomes close to that of random. Figure 6.21b shows the total collocation time for the same experiment. In terms of total collocation both balanced and random show similar performance.



(a) Beacons per Encounter.



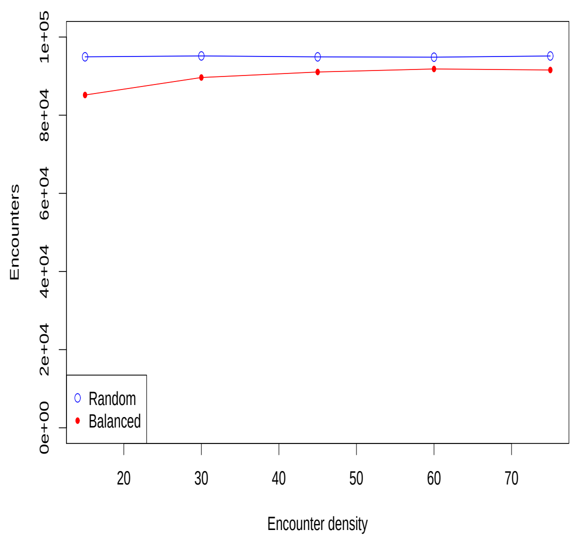
(b) Beacons per Duration.

Figure 6.20: Evaluation with random traces. Impact of variable network density.

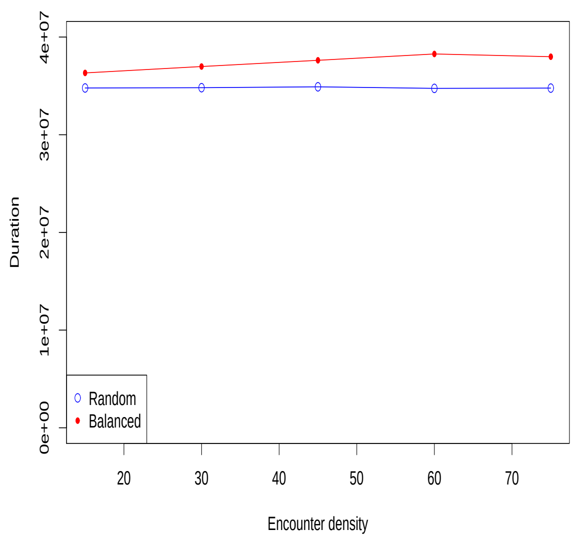
Figures 6.22a and 6.22b compare the beacon to encounter and beacon to encounter durations respectively. Both graphs show that balanced strategy required significantly less beacons compared to random without sacrificing performance. As opposed to real traces, the ratios remains the same throughout entire range of schedule rates due to uniform distribution of encounters throughout given time periods.

6.4.5 Adaptation Process

Figure 6.23a and Figure 6.24a illustrate the learning process by one of the nodes (ID 46) during the first 3 days of experiment with Reality Mining dataset. Figure 6.23a demonstrates the state of internal counters whereas Figure 6.24a shows the allocation of energy budget depending on available information about encounter distribution. At first, when there is no information about encounters the node starts by spreading the energy budget evenly throughout a day. As it learns about arrival patterns, it allocates more energy into busier timeslots. The counter values are updated once a day and take into account all past values of activity in the respective timeslots. The minimum F_{min} and maximum F_{max} beacon rates were set to 10% and 130% of normal (budgeted) beacon frequency. The convergence of an algorithm to a given temporal pattern was presented in Section 6.4.4.2 by measuring the performance of an algorithm and varying the rate of schedule changes.

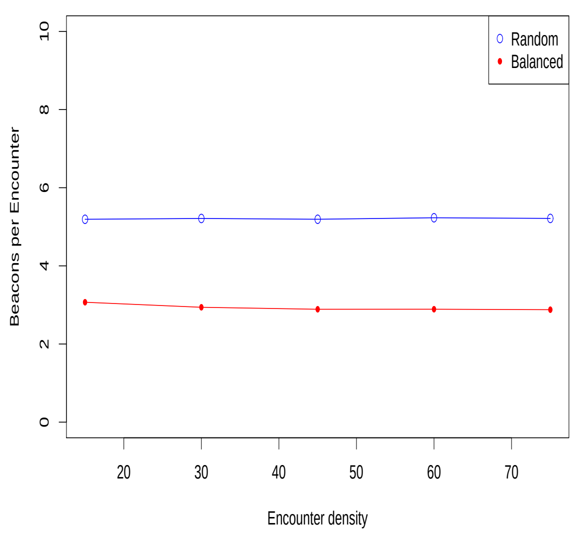


(a) Encounters.

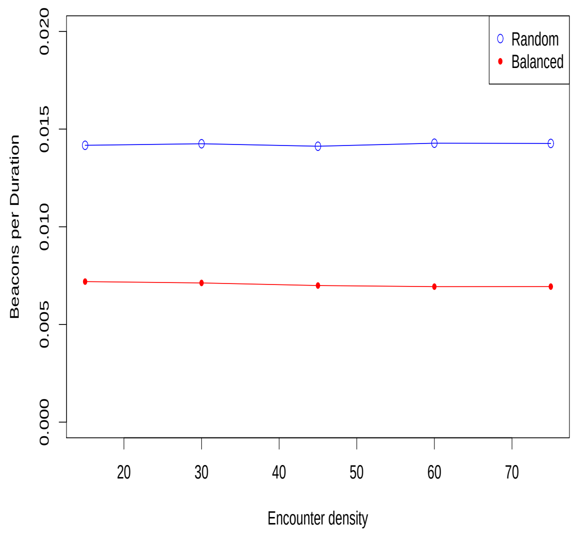


(b) Durations.

Figure 6.21: Evaluation with random traces. Impact of variable schedule change rate.



(a) Beacons per Encounter.



(b) Beacons per Duration.

Figure 6.22: Evaluation with random traces. Impact of variable schedule change rate.

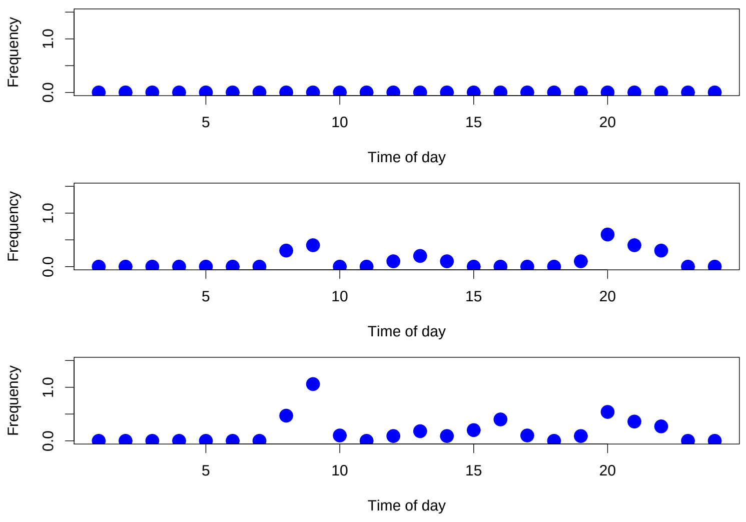


Figure 6.23: Illustration of a learning process. Counter values. The graph shows the counter values at successive days. The counter values represent the number of detected encounters in a given time slot.

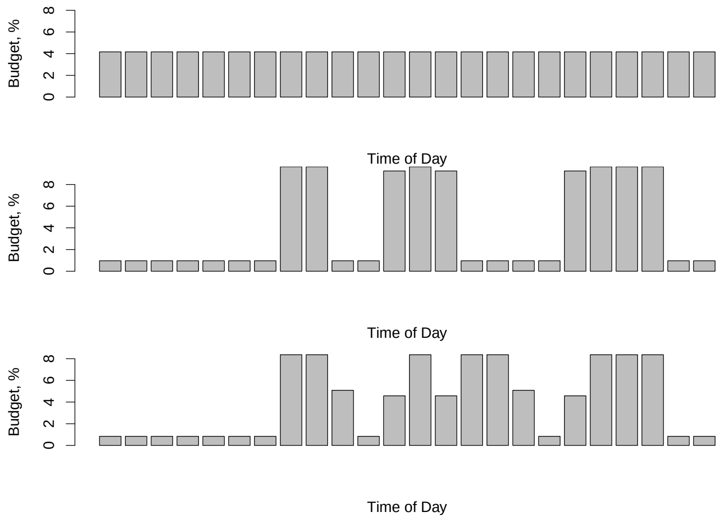


Figure 6.24: Illustration of a learning process. Budget distribution. The graph shows the budget values at successive days. The node starts by distributing the budget evenly and then puts more energy into active time slots.

6.5 Summary

In this chapter we have presented the evaluation of a macro level approach. The protocol works at a higher abstraction layer than micro-level protocol and controls the parameters of the underlying protocol by learning the temporal patterns. When working on top of low power listening medium access protocol, it can be used to control the frequency of power intensive discovery messages. For RFID tracking application, the macro level protocol controls the frequency of reader wake-up depending on time. The approach has been implemented in TinyOS-2.x, and simulated in Tossim environment using both synthetic and real mobility traces.

We have demonstrated that the adaptive scanning approach is able to reduce power consumption of mobile devices in the presence of temporal patterns. The approach is able to detect daily temporal patterns in the node connectivity and allocate the daily scanning budget dynamically according to the expected connectivity. We evaluated an approach on synthetic traces, with known and controllable parameters, which allowed testing the sensitivity of a protocol to connectivity density, level of dynamicity and various beacon rates. We also evaluated the approach on human and animal mobility traces, which reflected the real physical mobility patterns.

We evaluated our approach on 5 different data sets to make sure our approach adapts to various mobility scenarios. The data set from Intel Haggle and MIT Reality Mining experiments represented the physical proximity between individuals carrying Bluetooth devices for several days and months respectively. The traces from Dartmouth experiment represented the associations of mobile users (carrying laptops or PDAs) to static access points installed throughout a campus for over a year. Finally, the approach was evaluated on GPS data set representing the mobility of brushtail possums. A preliminary analysis confirmed the existence of temporal patterns in all data sets, with human mobility traces having stricter daily and weekly patterns.

The impact of adaptive scanning was evaluated on the performance of two applications: encounter tracking and data dissemination. As expected, adapting the scanning rate to a network temporal pattern allows the nodes to save energy while maintaining the network performance. In the encounter tracking application, the nodes were able to detect the same number of encounters while saving energy. In the data dissemination application, the network provided the same delivery rate, while consuming fewer scanning requests. The approach showed consistent behaviour over all data sets, although the performance over human data sets was higher. This result is expected, the human data sets had stricter daily patterns and more social interaction.

Chapter 7

Deployment

In this chapter we evaluate macro level object tracking approach for reducing RFID reader power consumption in wildlife tracking applications. The RFID tracking system has been designed as part of the WildSensing project [Dyo et al., 2009] [Dyo et al., 2010] to track the population of badgers in the Wytham Woods, Oxford, UK. WildSensing is an interdisciplinary project aimed at developing novel wireless tracking systems for large scale wildlife monitoring and tracking. The project is a collaboratie effort of Computer Laboratory at University of Cambridge, Computing Laboratory and Wildlife Conservation Research Unit (WildCRU) at University of Oxford with industrial support from WaveTrend Technologies LTD and Intel Corporation.

The chapter is organized as follows. We first provide a brief overview of RFID tracking system, the implementation of an approach including hardware design and deployment details. Next, we report the results of the deployment both indoors and outdoors by measuring the number of detections of RFID tags per unit of energy by the RFID node. We conclude the chapter by summarizing the results and experience from the deployment.

7.1 System Overview

The RFID tracking system is built from off-the-shelf commercial active RFID equipment and custom designed mote extension board, Figure 7.1. The advantage of using RFID tags is reliability, ubiquity and relatively low cost, which make it very suitable for large scale deployments. The system consists of four components:

- Active RFID tags attached to animals. The tags periodically emit RFID packets in the 433Mhz frequency band, with a period of 2.5s with a random offset to avoid potential tag synchronization and collisions. Each tag is hermetically sealed by an epoxy resin through a process called 'potting' to protect from humidity and physical damage. It is then physically attached to a collar, which is fixed around the animal neck during a

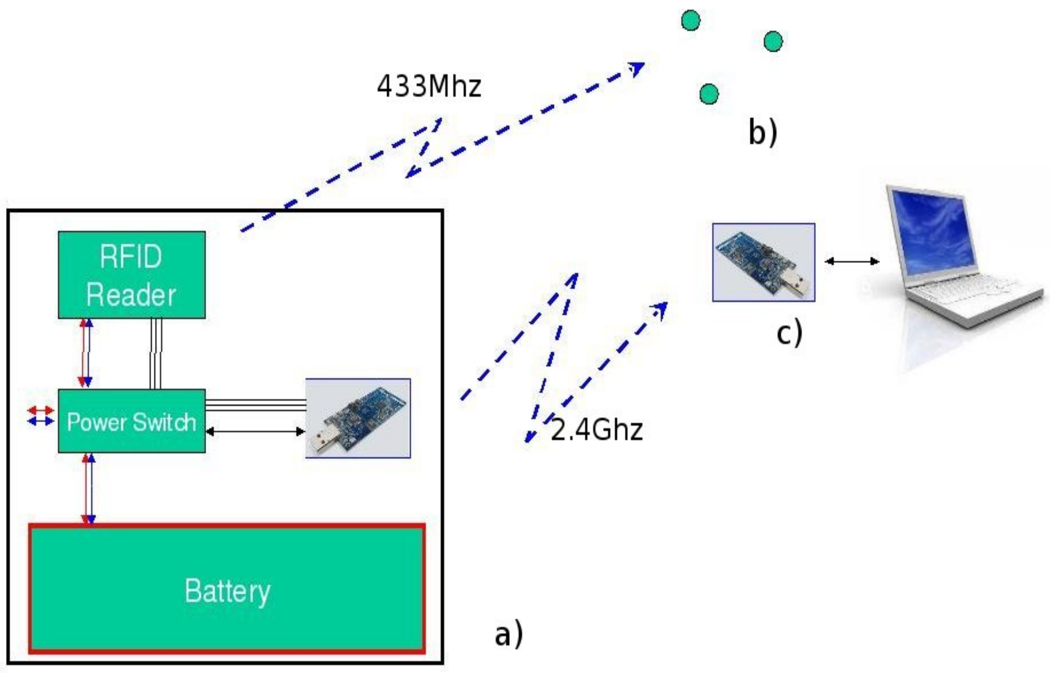


Figure 7.1: RFID node diagram: a) RFID node composed of Tmote Sky sensor node, custom-designed extension board, active RFID reader and a 12V battery. The extension board contains a voltage regulator, 2-port power switch and is controlled via the sensor node over I2C interface. b) Hermetically sealed RFID tags. c) a client software running on a laptop.

process called ’trapping’. Trapping is carried out several times a year and is usually a very labour intensive process. The tag lifetime depends on many factors, mainly battery capacity, beacon frequency, beacon duration and transmission power. The default settings provide a battery power of several years. The tag detection range depends on antenna type and radiation pattern, orientation, height, line of sight and position of readers and tags. The early field testing have shown that the detection range is also affected by an animal size and the positioning of the tag on the body. The currently used whip antenna detects tags attached to badgers within 10-30 meters.

- Active RFID nodes deployed throughout the forest. The location of readers is application specific, depending on the biological question. The current deployment will investigate how the neighbouring badger groups define territorial boundaries and how often individual species cross these boundaries. For this study, the readers have been deployed along the supposed boundaries, including burrows and latrines for several weeks.

  Each RFID node consists of WaveTrend LX201 active RFID reader, a TMote Sky sensor node and a custom designed mote extension board. The reader is connected to a serial port of the sensor node through a TTL-RS232 voltage converter on the extension board, which converts from TTL 3.3V level into RS232 12V level used by the reader. The extension board also regulates power from the 12V battery for up to two peripheral devices (sensors or actuators) including the reader. The output voltage ranges from 6V to 12V and is configurable either manually through potentiometers and switches on board or from the sensor node using a standard I2C interface. One power port is currently used to turn on and off the reader.

The reader operates in the auto-poll mode, in which it immediately provides the information about the detected tags on its serial interface. For each detected tag the reader provides the following information: received signal strength, tag serial number, type of tag, reader id and a checksum. The serial number increases by 1 per transmission and is needed for potential tag localization purposes, when the same animal is detected by several RFID nodes. When no tags are detected the reader sends an empty packet. More detailed specifications of a reader are provided in Table 7.1.

The sensor node controls the duty cycling of the reader: we have implemented the described reinforcement learning technique. The sensor node tunes the power to the RFID reader through an extension board by changing the output values of GPIO pins. During the learning period the reader is turned on and off on a periodic basis. Once the sensor node detects a pattern, it increases the wake up frequency for more active timeslots. It takes approximately 10s to boot a reader, which is also a minimum active interval for the reader. Once awake, the reader listens for tags for 300s and goes back to sleep. To take into account the short term information, the reader was configured to extend its active time by another 300s each time it detected a new tag.

The radio on the sensor node is duty cycled as well. The radio and the RFID reader use the same bus on the board, therefore one needs to be off while the other is using the bus. When the node is in logging mode, the reader uses the bus most of the time, while the radio is only turned on for a fraction of the time to listen to messages from nearby sensors. During the deployment the radio on detection node was turned on for 200ms every 5 seconds, resulting in 4% duty cycle of the radio. During the deployment the sensor node battery lasted for more than a month before being replaced. At that moment the sensor node is still operational, but as the battery voltage drops below a certain threshold (2.7V) it causes data corruption on Flash memory.

The Data Logger software was written in NesC for TinyOS operating system. The RFID

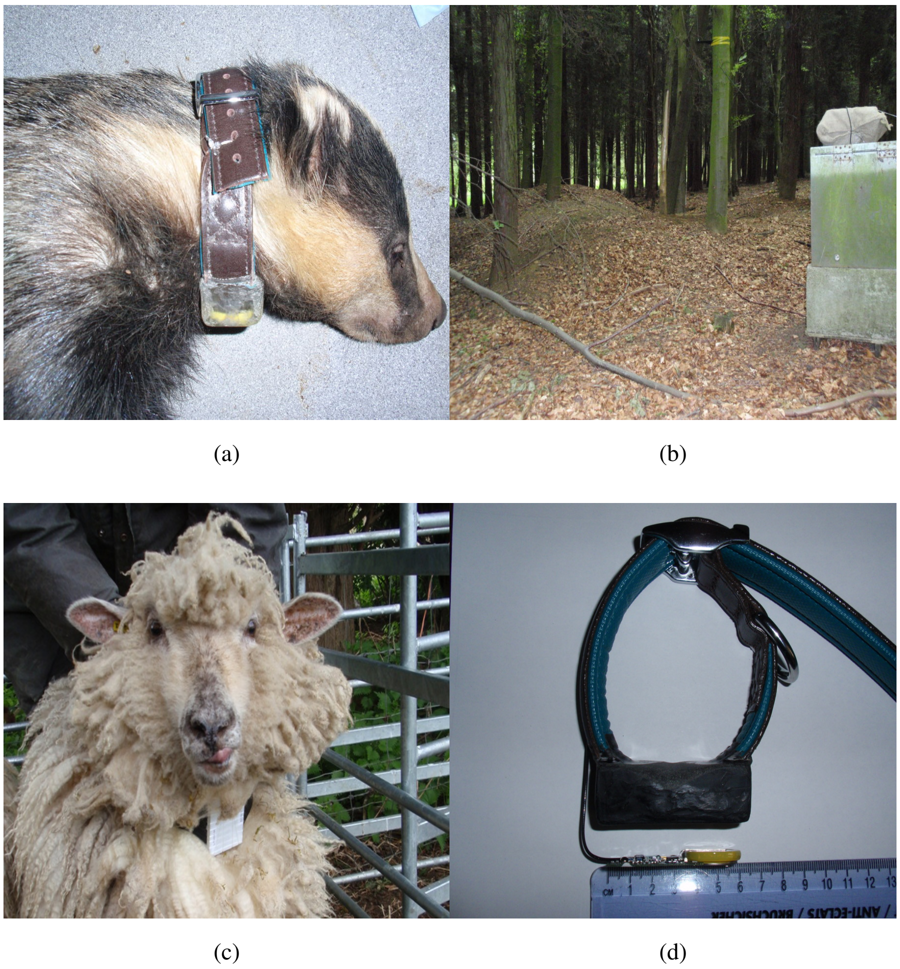


Figure 7.2: RFID system deployment. a) a tagged badger carrying a potted RFID tag. b) RFID node deployed near the badger burrow. c) RFID tagged sheep during early field tests. d) OEM-TM100 RFID active tag and a collar.

node logs the tag id, timestamp, received signal strength and serial number for each detected badger. The software also controls power to peripherals and can regulate voltage to peripheral devices and duty cycle the reader. The data logger monitors both sensor node and reader voltage and shuts down the system when the voltage becomes low. The RFID nodes are managed from the client software on the laptop, written in Java, and can take simple commands such as start and stop logging, set duty cycle, send summary and status information and upload the data.

The RFID nodes were put inside plastic containers to protect against rain and moisture and mounted at a certain height near the badger burrows and latrines.

- The client software, which is used to query the sensor node and download the data over wireless link. The client software is written in Java and can run on both Linux and Windows machines. The client software runs on a laptop and communicates with the RFID node, when the biologists visit the area around the RFID node. The client can be optionally connected via a multi-hop network. As shown by first deployments, RFID nodes produce very large amounts of data, for example, 5 tags can fill up 1MB of memory on the sensor node in less than a day. Even though the information is easily compressible, for example, by registering continuous readings as one event, the scientists insisted on keeping all the raw data. Firstly, the raw data are used by scientists for initial modeling of animal behavior. Secondly, storing raw data proved very important for debugging unexpected problems with hardware and bugs in the software. We noted, that in both cases the raw data are not required in real time. For this reason, the decision was made to store information locally, and periodically disseminate compact data summaries summarizing the badger activity in the last *t* minutes. The summary message contains the list of tags seen by the node in the last 3, 6, 12 and 24 hours.

The multi-hop network is used to send commands to the readers, check the reader status and disseminate highly compressed summary information about the recent tags from each reader. The status information is used for network management purposes to display general tag activity level in the network, storage and battery levels of RFID nodes.

The RFID nodes have been successfully tested both on students and wildlife animals. The first test has been done while tracking several graduate students in the Computer Science Department at University College London. The second series of tests were succesfully conducted on sheep, in a controlled but more realistic environment. These tests have shown in particular, the impact of body size on the signal absorption and consequently tag detection range. Finally, the system was deployed to track badgers in Wytham Woods, Oxford (Figure 7.2). Currently

Table 7.1: Technical specifications of the active RFID reader.

RF specification	**Value**
RF frequency	433Mhz
Demodulation	ASK
Sensitivity	-103dBm
Bandwidth	700kHz
Supply Voltage	6V-12V
Max current	90mA

Figure 7.3: The camera traps have been used to validate the detections and misses by RFID nodes. The data from the cameras have confirmed that badgers missing from RFID logs were missing indeed. Picture courtesy of Stephen Elwood, WildCRU, University of Oxford.

37 badgers have been tagged and 29 RFID nodes deployed in the woods. The target is to tag the entire population of badgers in the area, estimated at 200 animals.

As expected, the RFID readers were the largest source of power consumption in our system directly affecting system lifetime. The reader's power consumption in active state is rated at 90mA at a voltage level of 6V (Table 7.1). A typical node powered with a 12V 25Ah battery lasted for approximately 7.5 days during the field testing, so 29 node deployment required frequent changing of batteries. Therefore efficient energy management of RFID readers is critical for large-scale and long-term RFID system deployments.

7.2 Evaluation

In this subsection we present the initial results of the system deployment in Wytham Woods, Oxford, UK. The system running a macro level approach was run in parallel with non-duty cycling nodes and its performance was compared with ground-truth data. We consider the non-duty cycling nodes as producing ground-truth data, as they have been validated separately through photo and video cameras before and while being deployed in the forest (Figure 7.3). The data from the camera traps have been processed independently by a team of biologists

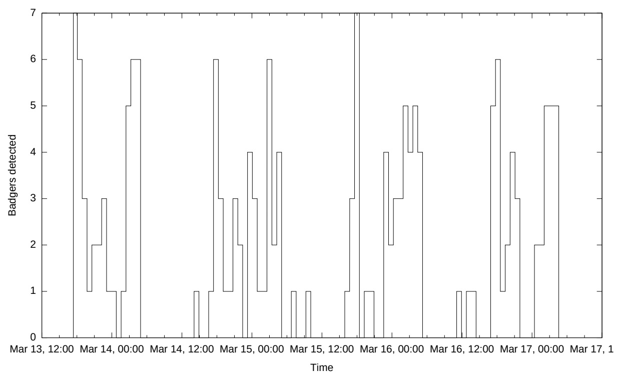


Figure 7.4: The histogram of a badger activity by time. The histogram shows that the badgers were most active at certain times during the night.

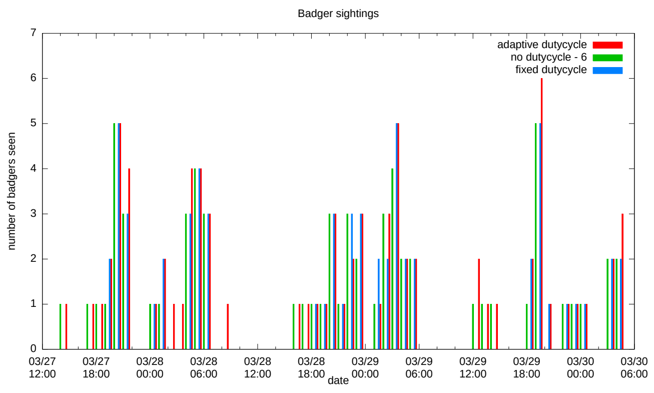


Figure 7.5: Graph showing the number of badgers seen by the detection nodes in 2 hour-intervals. Node Budget 50%.

and used to confirm the presence and absence of badgers in places where RFID nodes were deployed. All nodes in the experiment had the same hardware configuration and settings and were placed very close to each other. The adaptive duty cycling node had a budget setting of 50% in the first deployment and 30% in the following deployment. The values of budget settings were chosen to reflect the desired battery lifetime of the nodes.

We first collected data from non-duty cycling nodes and analyzed them for the presence of patterns. Figure 7.4 shows the activity distribution of badgers around one of the non-duty cycling nodes. It shows a pattern, where badgers are most active during the night and rarely come out during the day. The pattern has two large peaks with several small peaks repeating every day. According to zoologists the activity pattern depends on various factors including the light levels, micro-climate and changes every season.

In the next steps we compared the number of tags detected by non-duty cycling and duty

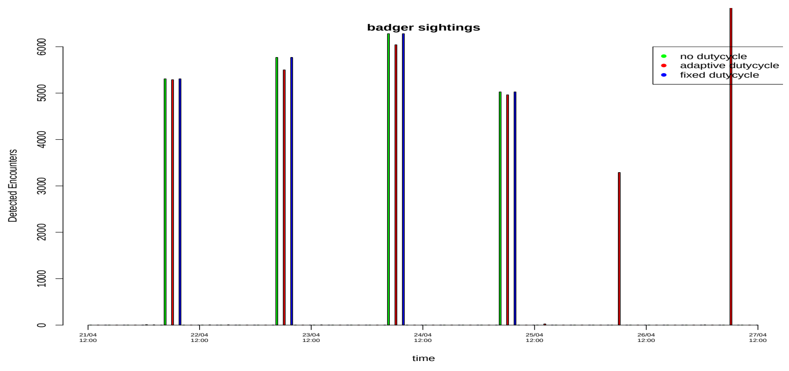


Figure 7.6: Graph showing the number of badgers seen by the detection nodes in 2 hour-intervals. Node Budget 30%.

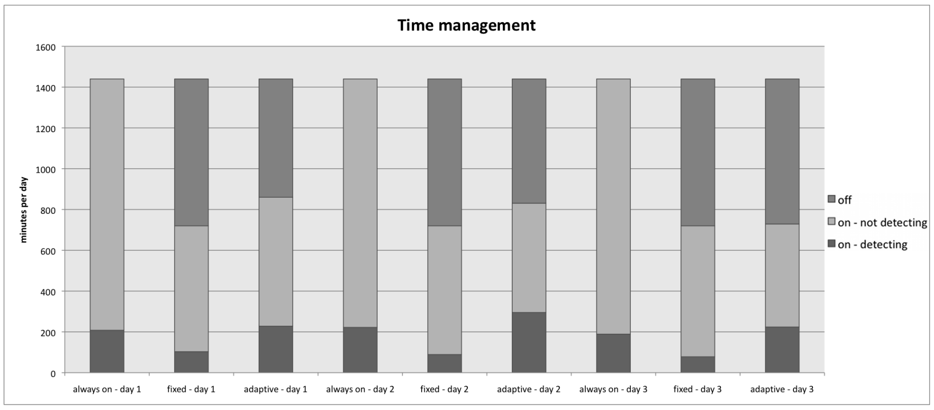


Figure 7.7: The amount of time the protocol spends being on and detecting, being on and not detecting, and being turned off for each day. Node Budget 50%.

cycling nodes as well as the amount of energy consumed by each of the nodes. The duty cycling node had a budget setting of 50%. The first graph (Figure 7.5) shows how many tags each node detected. The adaptive and the non-duty cycled node performed similarly, while the fixed duty cycle node missed some tags during the day. Despite the same hardware configuration and physical proximity the nodes sometimes reported different readings. For example, some tags seen by one of the nodes were not seen by the other and vice versa. This difference can be explained by a number of factors including slight differences in antenna orientation and battery levels of the nodes. The duty cycling node turned off the node periodically, and therefore maintained higher voltage levels on the battery, which might have impacted the sensitivity of the reader.

Figure 7.7 and Figure 7.8 show the amount of time the protocols spent in each of the following states: i) turned off, ii) turned on, but not detecting any tags (i.e., wasting energy),

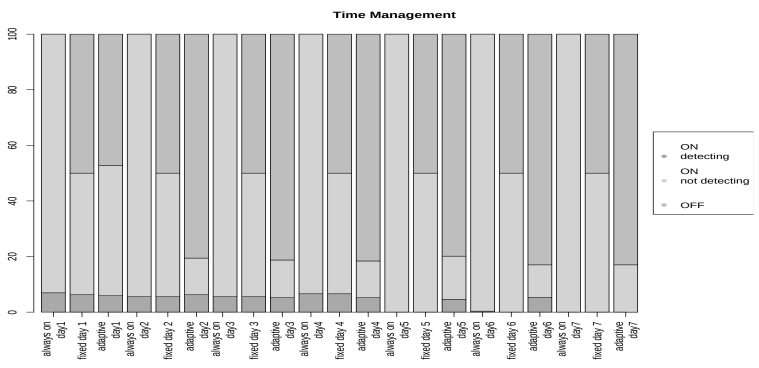


Figure 7.8: The amount of time the protocol spends being on and detecting, being on and not detecting, and being turned off for each day. Node Budget 30%.

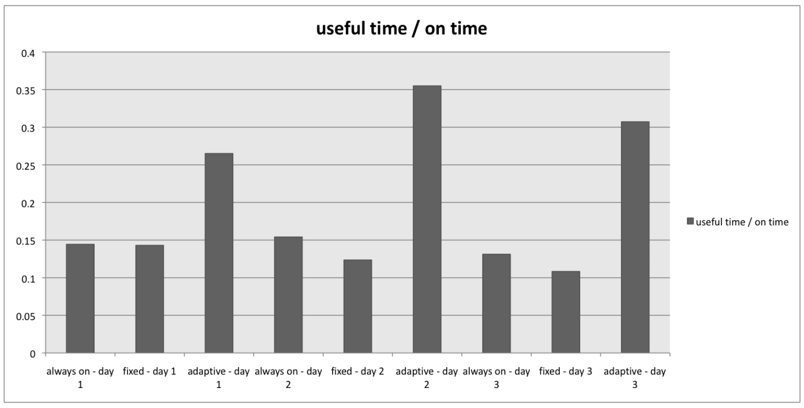


Figure 7.9: The amount of useful time the different protocols spend. Budget 50%

iii) turned on and detecting tags. We are trying to maximize the amount of sleep time, while minimizing the time the reader is on, but not detecting the tags. The non-duty cycled node detects tags only a fraction of its (always) on time. The fixed duty cycling node spends half of the time being turned off, and some of the remaining time detecting nodes. The adaptive protocol tries to learn the patterns of activity, therefore, initially it will waste some time trying to identify the active periods. As time goes on, it learns the patterns, and spends more time in the useful state.

To visualize our gain better, we also plotted on Figure 7.9 and Figure 7.10 the amount of *useful time* of each protocol. Useful time is defined as the ratio of being on while detecting to the total time the node was turned on. For the non-duty cycled node, it is rather low, since it never turns off, and spends only a very small fraction of the time detecting nodes. Although the fixed node spends some time being turned off, during the time when it is on, in only spends a relatively small amount of time detecting tags. Our protocol wakes up when necessary (i.e.,

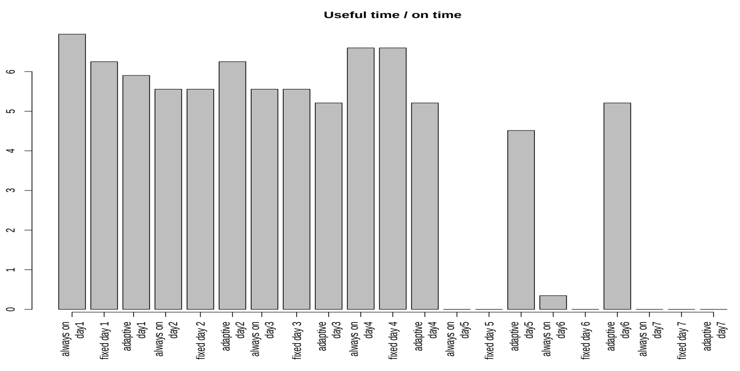


Figure 7.10: The amount of useful time the different protocols spend. Budget 30%

activity is detected), therefore increases the useful time of the node.

The results presented in this Chapter describe the system behaviour with at duty cycles of 30% and 50%. A more detailed analysis was done by replaying the raw data from the non-modified RFID readers in Tossim simulator, varying budget levels in Chapter 6, Section 6.1.

7.3 Summary

In this chapter we have described an evaluation of our approach by mini-deployment in the Computer Science Department of UCL tracking students for over a week, with one RFID node and 4 tagged students. The mini-deployment was followed by a real deployment in the Wytham Woods, UK and consisted of 29 RFID nodes deployed in Wytham Woods conservation area and 37 tagged badgers.

We first provided an overview of our system including the detailed description of the hardware and the software. For the evaluation purposes, the node running a macro level approach was run in parallel with normal nodes and detected similar number of detections per unit of energy consumed. The deployment has shown that the adaptive nodes are more efficient at detecting encounters than the nodes with fixed duty cycle. As a result of the deployment we have also found that differences in antenna orientation, environmental factors and battery levels can affect the results of the identical readers with the same hardware and software configuration.

To exclude the potential impact of differences in reader sensitivity and antenna, we replayed the traces from always-on reader to replay in Tossim simulator (in Chapter 6), varying the budget levels between 10% to 50% duty cycle. We found that the benefit in terms of detections per unit of energy is higher for lower duty cycles, which was consistent with deployment results.

Finally, the experience from deployment have confirmed the critical importance of duty cycling especially for large scale deployments and that the adaptive duty cycling is a viable approach to save energy in RFID detection systems.

Chapter 8

Conclusions

8.1 Summary

In this thesis we investigate an integrated approach for energy efficient node discovery in mobile wireless sensor networks. We look at the problem in a holistic way and provide an integrated two-layer approach to save power in mobile sensor networks. These two layers have access to different information and can make decisions at different levels of granularity. The micro layer deals with packet transmissions, advertising beacons at a certain rate and actual discovery of neighbouring nodes. At this layer we minimize idle listening time, overhearing and transmission costs. The micro layer does not make any assumptions about the mobility patterns and guarantees the node discovery and communication, whenever the nodes are in communication range with each other.

The macro layer may ’learn’ or ’know’ when the contacts are not expected and switches off expensive scanning during that period. The macro layer has been implemented as a standalone module, which registers encounters and calculates an optimal scanning rate for a given timeslot. We have presented an implementation of our approach and described a number of practical scenarios, where it could be applied. The scenarios range from partially-mobile networks, where static sinks need to detect fully mobile nodes, to fully mobile networks, where all nodes are mobile and need to save power while discovering each other.

In this chapter, we review the contributions of this thesis, provide a critical evaluation of our work and finally discuss potential future work.

8.2 Contributions

We summarize the contributions of the thesis.

8.2.1 Micro-level Scheduling

We have contributed by proposing a lightweight asynchronous wake-up scheduling protocol [Dyo and Mascolo, 2010]. The protocol allows to achieve ultra low duty cycling level of 0.3% while providing low latency in sparse and dense scenarios. In the protocol, each node follows an independent wake-up schedule, thus the network as a whole does not require time synchronization, which can be undesirable or unattainable in sensor networks. The nodes can, however, learn the schedules of other nodes and use a novel clock-skew estimation algorithm to compensate for relative clock drift between the nodes. The clock skew estimation uses very lightweight and computationally efficient recursive least squares linear regression estimation algorithm. A protocol has a novel schedule exchange extension, which further reduces the power consumption in dense scenarios. A schedule extension works by distributing the schedules of known nodes in every beacon packet, advertising not only its own but also the schedules of other nodes. This is useful in dense scenarios, to minimize the number of broadcasts (which need to be prepended by a preamble) and also in partially mobile scenarios, where sink learns and distributes the schedules of all mobile tags in the communication range. Finally, the protocol introduces a novel *clock quality metric*, which minimizes the clock aberration while transferring timestamp information through nodes with drifting clocks.

8.2.2 Macro-level Scheduling

Our contribution is an energy efficient node discovery protocol, which detects and exploits temporal regularities in node arrivals [Dyo and Mascolo, 2007, 2008]. We design a learning algorithm [Dyo and Mascolo, 2007] and apply it for partially mobile and fully mobile wireless sensor networks. In a partially mobile sensor networks, the algorithm aims at maximizing the number of mobile node detections given sink's daily energy budget. In fully mobile wireless networks [Dyo and Mascolo, 2008], the algorithm tries to increase the number of encounters between the nodes given a fixed energy budget. The algorithm uses a simple reinforcement learning technique [Kaelbling et al., 1996], it is light-weight and suitable for resource constrained sensor nodes. Our approach is different from other protocols described in the literature by exploiting temporal patterns in the mobility of nodes. Another advantage of an approach is that it does not require a domain knowledge and can learn the patterns online.

The macro-level scheduling is designed to control the parameters of underlying micro-level scheduler by adjusting the beacon frequencies or awake time of the radio. Therefore, the two techniques compliment each other at different layers of granularity. The macro-level technique takes into account long term patterns and makes decisions at a higher level whether to expect a contact. A micro-level technique tries to reduce awake time while expecting a certain encounter.

We evaluated our approach through simulations, implementation on sensor nodes and a deployment. The simulation was conducted with a number of real human and animal mobility traces and showed an advantage of an adaptive approach. The implementation on sensor nodes provides an API to control the duty cycle, assign energy budget and facilitate the neighbour management process.

Finally, we successfully applied the approach for adaptive duty cycling of active RFID reader in a wildlife tracking application for tracking badgers in Wytham Woods, UK. A working implementation has been deployed to track wildlife animals and shown to detect more encounters per unit of energy. We replayed the the raw data in simulations to experiment with various energy budget settings and exclude potential differences between the individual readers. The deployment has shown that the balanced approach reduces power consumption and outperforms fixed duty cycling.

8.3 Critical Evaluation

The goal of the thesis was to investitate an adaptive duty cycling approach, which is low power, adaptive to temporal patterns and suitable for resource-constrained sensor nodes. We evaluate our approach with respect to these requirements.

In Chapter 5 we have demonstrated that the protocol achieves very low duty cycles through measurements on a small-scale testbed. The protocol has been implemented for CC2420 radio on Tmote Sky sensor nodes under TinyOS [Levis, 2006] operating system. We evaluated the protocol by measuring the power consumption on sender and a receiver for different network densities under various traffic loads. The protocol provides duty cycling of as low as 0.3%, which is comparable with self-discharge rate of lithium-ion batteries[1].

In Chapter 6 we have demonstrated through real human and animal mobility traces that our approach is adaptive to temporal mobility patterns. The approach is able to learn the daily patterns and reallocate the daily budget accordingly. Adaptation mechanism is useful when nodes have dynamic activity patterns, caused by seasonal changes or differences in activity levels between individual nodes. In both cases, the application designers do not have to hardwire the node activity pattern information into the application: the algorithm should be able to learn the pattern and adjust the duty cycle accordingly. The efficiency of an approach depends on the predictability in node activity. For completely random activity distribution, the efficiency is the

[1]The self-discharge rate of Lithium batteries is less than 10% per year resulting in 10 years shelf-life[Kiehne, 2003]. An equivalent duty cycle for a TMote Sky to reach 10 years on 2500Ah Lithium battery is 0.1%.

same as the random scanning. The approach gets more efficient as encounter pattern becomes more predictable. These temporal patterns can be observed in a histogram of all traces we used in the evaluation. The traces represent human and animal mobility in various settings: the students with laptops roaming around the campus, people carrying mobile phones in every day life, and students carrying Bluetooth iMotes walking around the city. In case of animal traces, the traces represent the mobility of wild possums equipped with GPS tags. In all of these traces, the nodes had a temporal pattern, which could be slowly changing due to seasonal variations.

The approach was developed for memory and resource constrained wireless sensors, therefore it has to have a small memory footprint and be efficient in operation. The implementation of a macro-level approach takes only 16kb of ROM and 2kB of RAM, which means it is feasible for a wide range of resource constrained sesors available today.

8.4 Future work

The work in the thesis has identified a number of interesting research questions. We present the most promising directions below.

8.4.1 Algorithm Improvement

The balanced algorithm used in macro-level scheduling approach could be improved to predict the time of specific encounters instead of predicting an overall encounter frequency. This would allow making more fine grained scanning decisions and further optimize the resource usage. The first step towards such an algorithm is a detection of strictly periodic components in the node connectivity patterns. The algorithm could leverage from existing work on periodic pattern detection in digital signal processing, data mining and statistical data analysis. Another area for potential improvement is dynamic budget assignment depending on the node activity and current battery level. In current macro layer approach, all nodes have the same fixed budget. However, not all nodes have the same battery level or level of activity. For example, some sinks might be equipped with solar panels, which recharge their batteries. With dynamic budget assignment, resource rich nodes can automatically assume the role of scanners discovering the mobile nodes in the vicinity. In the same vein, the nodes with higher activity level can reduce their budget while keeping the same level of connectivity. Finally, the interesting direction is to merge information from all available sensors, such as accelerometers, passive infrared and radio to infer the current context and calculate an optimal level of the duty cycling.

8.4.2 Routing Integration

An interesting question is related to the interdependency of routing and duty cycling. The duty cycling process actively changes the topology of the sparsely connected mobile network. As a result of sampling, or duty cycling, the nodes will see only the subset of real communication opportunities. A first step would be to take into account the variance in periodicity and extracts the 'stable' portions of the network as an extended temporal graph. The graph can be used directly by scheduling algorithms similar to transport scheduling algorithms to route packets in the temporal network. For example, if the time of a future encounter with a destination node is known with a certain degree of confidence, it could be used to prevent unnecassary forwarding or replicating the message. This could also be seen as an extension of practical routing algorithm for opportunistic networks [Jones et al., 2005] by integrating routing with duty cycling.

8.4.3 MAC protocol

Dynamic and varying connectivity is one of the challenging problems in mobile networks. An interesting future work is deploying the micro-level protocol for real mobile applications and measuring its performance. A long-term deployment for wildlife tracking and monitoring applications, where tiny sensors attached to animals interact with each other to monitor and sense an environment would be the final validation of the work in this thesis. Further, it will open up many other important issues. The development of new radios and ultra low power wake-up circuits, integration with application could lead to an interesting future work.

Bibliography

Adaptive Sampling Designs. URL http://www.eecs.umich.edu/ qstout/AdaptSample.html. Accessed August 31, 2009.

Adaptive Sampling and Prediction. A DOD/ONR Multi-disciplinary University Research Initiative. URL http://www.princeton.edu/ dcsl/asap. Accessed August 31, 2009.

Engineering Research Centre for Collaborative Adaptive Sensing of the Atmosphere (CASA). URL http://casa.ece.uprm.edu. Accessed August 31, 2009.

Steve Alpern. The Rendezvous Search Problem. *SIAM Journal of Control Optimization*, 33(3): 673–683, 1995. ISSN 0363-0129.

Shah Bhatti, James Carlson, Hui Dai, Jing Deng, Jeff Rose, Anmol Sheth, Brian Shucker, Charles Gruenwald, Adam Torgerson, and Richard Han. MANTIS OS: an Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms. *Mobile Networks and Applications*, 10:563–579(17), January 2005.

Steven A. Borbash, Anthony Ephremides, and Michael J. McGlynn. an Asynchronous Neighbor Discovery Algorithm for Wireless Sensor Networks. *Ad Hoc Networks*, 5(7):998–1016, 2007. ISSN 1570-8705.

George Edward Pelham Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1994. ISBN 0130607746.

Joel Branch, Boleslaw Szymanski, Chris Giannella, Ran Wolff, and Hillol Kargupta. In-Network Outlier Detection in Wireless Sensor Networks. In *ICDCS'06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, page 51, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2540-7.

Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks. In *SenSys'06: Proceedings of* *the 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-343-3.

Nicolo Cesa-Bianchi and Paul Fischer. Finite-Time Regret Bounds for the Multiarmed Bandit Problem. In *ICML'98: Proceedings of the 15th International Conference on Machine Learning*, pages 100–108. Morgan Kaufmann, 1998.

A. Chakrabarti. Exploiting Predictable Observer Mobility for Power Efficient Sensor Network Communication. 2003.

Sravanthi Chalasani and James Conrad. A Survey of Energy Harvesting Sources for Embedded Systems. In *Southeastcon, 2008. IEEE*, pages 442–447, 2008.

Supriyo Chatterjea and Paul Havinga. An Adaptive and Autonomous Sensor Sampling Frequency Control Scheme for Energy-Efficient Data Acquisition in Wireless Sensor Networks. In *DCOSS'08: Proceedings of the 4rd IEEE International Conference on Distributed Computing in Sensor Systems*, Santorini, Greece, June 2008.

Benjie Chen, Kyle Jamieson, Hari Balakrishnan, and Robert 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*, pages 85–96, New York, NY, USA, 2001. ACM Press. ISBN 1-58113-422-3.

David Chu, Amol Deshpande, Joseph M. Hellerstein, and Wei Hong. Approximate Data Collection in Sensor Networks using Probabilistic Models. In *ICDE'06: Proceedings of the 22nd International Conference on Data Engineering*, page 48, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2570-9.

Michael D. Colagrosso. Intelligent broadcasting inmobile ad hoc networks: three classes of adaptive protocols. *EURASIP Journal of Wireless Communication Networking*, 2007(1):25–25, 2007. ISSN 1687-1472.

Costis Coumpis and Simon Aliwell. Energy Harvesting Technologies to Enable Remote and Wireless Sensing. Sensors and Instrumentation. Knowledge Transfer Network, June 2008.

Todd Dennis, W.C. Chen, I.M. Koefoed, C.J. Lacoursiere, and M.M. Walker. Performance Characteristics of Small Global Positioning System Collars. Technical report, School of Biological Sciences, University of Auckland, Private Bag 92019, Auckland Mail Centre, Auckland 1142, New Zealand, 2008.

L. Doherty, B.A. Warneke, B.E. Boser, and K.S.J. Pister. Energy and Performance Considerations for Smart Dust. In *International Journal of Parallel Distributed Systems and Networks*, volume 4, pages 121–133, 2001.

Christl A. Donnelly, Rosie Woodroffe, D. R. Cox, F. John Bourne, C. L. Cheeseman, Richard S. Clifton-Hadley, Gao Wei, George Gettinby, Peter Gilks, Helen Jenkins, W. Thomas Johnston, Andrea M. Le Fevre, John P. McInerney, and W. Ivan Morrison. Positive and Negative Effects of Widespread Badger Culling on Tuberculosis in Cattle. *Nature*, 439:pp. 843–846, 2006.

Catalin Drula, Cristiana Amza, Franck Rousseau, and Andrzej Duda. Adaptive Energy Conserving Algorithms for Neighbor Discovery in Opportunistic Bluetooth Networks. *IEEE Journal on Selected Areas in Communications*, 25(1):96–107, 2007.

Shu Du, Amit Kumar Saha, and David B. Johnson. RMAC: A Routing-Enhanced Duty-Cycle MAC Protocol for Wireless Sensor Networks. In *INFOCOM'07: Proceedings of the 26th Annual IEEE Conference on Computer Communications*, pages 1478–1486, May 2007.

Jay C. Dunlap, Jennifer J. Loros, and Patricia J. DeCoursey. *Chronobiology: Biological Timekeeping*. Sinauer Associates, Sunderland, MA, 2004. ISBN 087893149X.

Prabal Dutta and David Culler. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *SenSys'08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 71–84, New York, NY, USA, 2008. ACM Press. ISBN 978-1-59593-990-6.

Prabal Dutta, David Culler, and Scott Shenker. Asynchronous Neighbor Discovery: Finding Needles of Connectivity in Haystacks of Time. In *IPSN'08: Proceedings of the 7th international conference on Information processing in sensor networks*, pages 531–532, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3157-1.

Vladimir Dyo and Cecilia Mascolo. A Node Discovery Service for Partially Mobile Sensor Networks. In *MIDSENS'07: Proceedings of IEEE International Workshop on Sensor Network Middleware*, Newport Beach, California, USA, November 2007. IEEE Computer Society.

Vladimir Dyo and Cecilia Mascolo. Efficient Node Discovery in Mobile Wireless Sensor Networks. In *DCOSS'08: Proceedings of the 4rd IEEE International Conference on Distributed Computing in Sensor Systems*, Santorini, Greece, June 2008.

Vladimir Dyo and Cecilia Mascolo. MAC*ron* - Energy-efficient Wake-up Scheduling for Mobile Sensor Networks. Technical report, Department of Computer Science, University College London. Submitted for publication, 2010.

Vladimir Dyo, Stephen Ellwood, Anders Lindgren, Mike Lonerghan, David Macdonald, Andrew Markham, Cecilia Mascolo, Bence Pasztor, Niki Trigoni, and Ricklef Wohlers. WildSensing: A Hybrid Framework of Mobile and Sensor Nodes for Wildlife Monitoring. Wildlife Monitoring with Sensor Technology workshop, Cambridge, UK, 2009.

Vladimir Dyo, Stephen Ellwood, Anders Lindgren, David Macdonald, Andrew Markham, Cecilia Mascolo, Bence Pasztor, Niki Trigoni, and Ricklef Wohlers. A Lazy Approach to Wildlife Monitoring: Collect Everything, All in Good Time. To be submitted to IPSN, 2010.

Nathan Eagle and Alex Pentland. Eigenbehaviors: Identifying Structure in Routine. In *Behavioral Ecology and Sociobiology*, volume 63:7, pages 1057–1066, 2009.

Nathan Eagle and Alex (Sandy) Pentland. Reality Mining: Sensing Complex Social Systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006. ISSN 1617-4909.

Dieter Ebert, Klaus P. Ebmeier, T. Rechlin, and Wolfgang R. Kaschka. Biological Rhythms and Behavior, Advances in Biological Psychiatry. ISSN 0378-7354, 1983.

Donald Patrick Eickstedt. Adaptive Sampling in Autonomous Marine Sensor Networks. PhD thesis. Joint Program in Oceanography/Applied Ocean Science and Engineering (Massachusetts Institute of Technology, Dept. of Mechanical Engineering; and the Woods Hole Oceanographic Institution), 2006.

Amre El-Hoiydi and Jean-Dominique Decotignie. WiseMAC: an Ultra Low Power MAC Protocol for the Downlink of Infrastructure Wireless Sensor Networks. In *ISCC'04: Proceedings of the Ninth International Symposium on Computers and Communications 2004 Volume 2 (ISCC'04)*, pages 244–251, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7803-8623-X.

Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-grained Network Time Synchronization Using Reference Broadcasts. *SIGOPS Operating Systems Review*, 36(SI):147–163, 2002. ISSN 0163-5980.

Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. Timing-sync Protocol for Sensor Networks. In *SenSys'03: Proceedings of the 1st International Conference on Embedded*

Networked Sensor Systems, pages 138–149, New York, NY, USA, 2003. ACM Press. ISBN 1-58113-707-9.

Gray Girling, Jennifer Li, Kam Wa, and Paul Osborn. The Design and Implementation of a Low Power Ad Hoc Protocol Stack. In *IEEE Personal Communications*, pages 8–15, 2000.

John C. Gittins. Bandit Processes and Dynamic Allocation Indices. *Journal of the Royal Statistical Society, Series B, 41*, pages 148–177, 1979.

Lin Gu and John A. Stankovic. Radio-Triggered Wake-Up for Wireless Sensor Networks. *Real-Time Systems*, 29(2-3):157–182, 2005. ISSN 0922-6443.

Zygmunt J. Haas, Joseph Y. Halpern, and Li Li. Gossip-based Ad Hoc Routing. *IEEE/ACM Transactions on Networking*, 14(3):479–491, 2006. ISSN 1063-6692.

Douglas Herbert, Gaspar Modelo Howard, Carlos Perez-Toro, and Saurabh Bagchi. Fault Tolerant ARIMA-based Aggregation of Data in Sensor Networks. In *IEEE International Conference on Dependable Systems and Networks*, 2007.

Barbara Hohlt, Lance Doherty, and Eric Brewer. Flexible Power Scheduling for Sensor Networks. In *IPSN'04: Proceedings of the third International Symposium on Information Processing in Sensor Networks*, pages 205–214, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-846-6.

Jason Hsu, Sadaf Zahedi, Aman Kansal, Mani Srivastava, and Vijay Raghunathan. Adaptive Duty Cycling for Energy Harvesting Systems. In *ISLPED'06: Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, pages 180–185, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-462-6.

Feiyi Huang and Yang Yang. Energy Efficient Collision Avoidance MAC Protocol in Wireless Mesh Access Networks. In *IWCMC'07: Proceedings of the 2007 International Conference on Wireless Communications and Mobile Computing*, pages 272–277, New York, NY, USA, 2007. ACM Press. ISBN 978-1-59593-695-0.

Texas Instruments. CC1100 Low Power Sub 1-Ghz RF Transceiver. Technical Datasheet (Rev. E), 2008.

International Organization for Standardization. ISO/IEC 18000-1:2008, 2008.

Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a Delay Tolerant Network. In *SIGCOMM'04: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 145–158, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-862-8.

Jehn-Ruey Jiang, Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Hwang Lai. Quorum-based Asynchronous Power-saving Protocols for IEEE 802.11 Ad Hoc Networks. *Mobile Networking Applications*, 10(1-2):169–181, 2005. ISSN 1383-469X.

David B. Johnson and David A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

Evan P. C. Jones, Lily Li, and Paul A. S. Ward. Practical Routing in Delay-Tolerant Networks. In *WDTN'05: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, pages 237–243, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-026-4.

Philo Juang, Hidekazu Oki, and Yong Wang. Energy Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *ASPLOS'02: Proceedings of the 7th International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM Press, October 2002.

Hyewon Jun, Mostafa H. Ammar, Mark D. Corner, and Ellen W. Zegura. Hierarchical Power Management in Disruption Tolerant Networks with Traffic-aware Optimization. In *CHANTS'06: Proceedings of the SIGCOMM Workshop on Challenged Networks*, pages 245–252, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-572-X.

Leslie Pack Kaelbling, Michael L. Littman, and Andrew P. Moore. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

Abtin Keshavarzian, Huang Lee, and Lakshmi Venkatraman. Wakeup Scheduling in Wireless Sensor Networks. In *MobiHoc'06: Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 322–333, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-368-9.

Heinz Albert Kiehne. *Battery Technology Handbook*. CRC Press, 2003. ISBN 0824742494.

David Kotz and Kobby Essien. Analysis of a Campus-Wide Wireless Network. *Wireless Networks*, 11(1-2):115–133, January 2005. ISSN 1022-0038.

Evangelos Kranakis, Danny Krizanc, and Sergio Rajsbaum. Mobile agent rendezvous: A survey. In P. Flocchini and L. Gasieniec, editors, *SIROCCO'06: Proceedings of 13th International Colloquium on Structural Information and Communication Complexity*, volume 4056 of *LNCS*, pages 1–9. Springer-Verlag Berlin Heidelberg, July 2006.

Andreas Krause and Carlos Guestrin. Nonmyopic Active Learning of Gaussian Processes: an Exploration-Exploitation Approach. In *ICML'07: Proceedings of the 24th International Conference on Machine Learning*, pages 449–456, New York, NY, USA, 2007. ACM Press. ISBN 978-1-59593-793-3.

Gerben Kuijpers, Thomas Toftegaard Nielsen, and Ramjee Prasad. Optimizing Neighbor Discovery for Ad Hoc Networks Based on the Bluetooth PAN Profile. *Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on*, 1:203–207, 2002.

Philip Alexander Levis. TinyOS: An Open Operating System for Wireless Sensor Networks (Invited Seminar). In *MDM'06: Proceedings of the 7th International Conference on Mobile Data Management*, page 63, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2526-1.

En-Yi Lin. A Comprehensive Study of Power Efficient Rendezovous Schemes for Wireless Sensor Networks. PhD thesis. Computer Science Department, University of Berkeley, 2005.

Anders Lindgren, Avri Doria, and Olov Scheln. Probabilistic Routing in Intermittently Connected Networks. *SIGMOBILE Mobile Computation and Communication Review*, 7(3):19–20, July 2003.

Zhenzhen Liu and Itamar Elhanany. RL-MAC: a Reinforcement Learning Based MAC Protocol for Wireless Sensor Networks. *International Journal of Sensor Networks*, 1(3/4):117–124, 2006. ISSN 1748-1279.

Gang Lu, Bhaskar Krishnamachari, and Cauligi Raghavendra. an Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks. In *International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, Santa Fe, NM, April 2004.

Geoffrey Mainland, David C. Parkes, and Matt Welsh. Decentralized, Adaptive Resource Allocation for Sensor Networks. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 315–328, Berkeley, CA, USA, 2005. USENIX Association.

Miklos Maroti, Branislav Kusy, Gyula Simon, and Akos Ledeczi. The Flooding Time Synchronization Protocol. In *SenSys'04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 39–49, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-879-2.

Yutaka Matsuo, Naoaki Okazaki, Kiyoshi Izumi, Yoshiyuki Nakamura, Takuichi Nishimura, Kiti Hasida, and Hideyuki Nakashima. Inferring Long-term User Property based on Users' Location History. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), (2007.1)*, 2007.

Michael J. McGlynn and Steven A. Borbash. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *MobiHoc'01: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 137–145, New York, NY, USA, 2001. ACM Press. ISBN 1-58113-428-2.

David Mech and Shannon Barber. A Critique of Wildlife Radio-tracking and its Use in National Parks: a Report to the U.S. National Park Service, 2002.

MoteIV. TMote Sky Datasheet. MoteIV Corporation, 2006.

John Moy. OSPF version 2. RFC 2328, Internet Engineering Task Force, April 1998.

M. Musolesi, S. Hailes, and C. Mascolo. Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks. In *Proceedings of the IEEE 6th International Symposium on a World of Wireless, Mobile, and Multimedia Networks (WoWMoM 2005)*, Taormina, Italy, Jun 2005. IEEE Press.

Esmaeil Shahrak Nadimi, Henning T. Sogaard, Thomas Bak, and Frank W. Oudshoorn. ZigBee-based Wireless Sensor Networks for Monitoring Animal Presence and Pasture Time in a Strip of New Grass. *Computers and Electronics in Agriculture*, 61(2):79–87, 2008. ISSN 0168-1699.

Suman Nath and Phillip B. Gibbons. Communicating via Fireflies: Geographic Routing on Duty-cycled Sensors. In *IPSN '07: Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, pages 440–449, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-638-X.

Tomasz Naumowicz, Robin Freeman, Andreas Heil, Martin Calsyn, Eric Hellmich, Alexander Brändle, Tim Guilford, and Jochen Schiller. Autonomous Monitoring of Vulnerable Habitats

Using a Wireless Sensor Network. In *REALWSN'08: Proceedings of the Workshop on Real-world Wireless Sensor Networks*, pages 51–55, New York, NY, USA, 2008. ACM Press. ISBN 978-1-60558-123-1.

Jeffrey J. Pan, Qiang Yang, and Sinno Jialin Pan. Online Co-localization in Indoor Wireless Networks by Dimension Reduction. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, Vancouver, Canada, 2007.

Panasonic. MP Motion Sensor NaPiOn, Part number: AMN42121 (slight motion detection type with low current consumption) Panasonic Electric Works Co. Ltd, 2009.

Charles E. Perkins and Elizabeth M. Royer. Ad-hoc On-Demand Distance Vector Routing. In MILCOM'97 panel on Ad Hoc Networks, November 1997.

Marina Petrova, Lili Wu, Petri Mahonen, and Janne Riihijarvi. Interference Measurements on Performance Degradation between Colocated IEEE 802.11g/n and IEEE 802.15.4 Networks. In *ICN'07: Proceedings of the 6th International Conference on Networking*, page 93, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2805-8.

Nathan Pletcher. Ultra-Low Power Wake-Up Receivers for Wireless Sensor Networks. PhD thesis. University of California, Berkeley, 2008.

Joseph Polastre, Jason Hill, and David Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *SenSys'04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 95–107, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-879-2.

G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. *Commun. ACM*, 43(5): 51–58, 2000. ISSN 0001-0782.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2007. ISBN 3-900051-07-0.

J. Rabaey, J. Ammer, J. L. da Silva Jr., and D. Patel. PicoRadio: Ad-Hoc Wireless Networking of Ubiquitous Low-Energy Sensor/Monitor Nodes. In *WVLSI'00: Proceedings of the IEEE Computer Society Annual Workshop on VLSI*, page 9, Washington, DC, USA, 2000. IEEE Computer Society. ISBN 0-7695-0534-1.

V. Raghunathan, C. Schurgers, S. Park, and Mani B. Srivastava. Energy aware wireless microsensor networks. In *IEEE Signal Processing Magazine*, volume 19, pages 40–50, march 2002.

Mohammad Rahimi, Richard Pon, William J. Kaiser, Gaurav S. Sukhatme, and Deborah Estrin. Adaptive Sampling for Environmental Robotics. In *IEEE International Conference on Robotics and Automation*, pages 3537–3544, 2004.

Injong Rhee, Ajit Warrier, Mahesh Aia, and Jeongki Min. Z-MAC: a Hybrid MAC for Wireless Sensor Networks. In *SenSys'05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 90–101, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-054-X.

Herbert Robbins. Some Aspects of the Sequential Design of Experiments. In *Bulletin of the American Mathematical Society*, volume 55, page 527535, 1952.

George Roussos. *Networked RFID: Systems, Software and Services*. Springer Publishing Company, Incorporated, 2008. ISBN 1848001525, 9781848001527.

Nicholas Roy and Gregory Dudek. Collaborative Exploration and Rendezvous: Algorithms, Performance Bounds and Observations. *Autonomous Robots*, 11(2):117–136, September 2001.

SAPHE-Project. SAPHE Architecture Overview. Smart and Aware Pervasive , January 2008.

Yoav Sasson, David Cavin, and André Schiper. Probabilistic Broadcast for Flooding in Wireless Mobile Ad hoc Networks. In *WCNC'03: Proceedings of IEEE Wireless Communications and Networking Conference*, March 2003.

James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD Trace Set Cambridge/haggle/imote (v. 2006-01-31). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle/imote, January 2006.

Victor Shnayder, Bor rong Chen, Konrad Lorincz, Thaddeus R. F. Fulford Jones, and Matt Welsh. Sensor Networks for Medical Care. In *SenSys'05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 314–314, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-054-X.

Pavan Sikka, Peter Corke, Philip Valencia, Christopher Crossman, Dave Swain, and Greg Bishop-Hurley. Wireless Adhoc Sensor and Actuator Networks on the Farm. In *IPSN'06:*

Proceedings of the third International Symposium on Information Processing in Sensor Networks, pages 492 – 499, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-334-4.

Roger W. Sinnott. Virtues of the Haversine. *Sky and Telescope*, 68:158, December 1984.

Tara Small and Zygmunt J. Haas. The Shared Wireless Infostation Model: a New Ad Hoc Networking Paradigm (or Where There is a Whale, There is a Way). In *MobiHoc'03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 233–244, New York, NY, USA, 2003. ACM Press. ISBN 1-58113-684-6.

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

Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and Wait: an Efficient Routing Scheme for Intermittently Connected Mobile Networks. In *WDTN'05: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Metworking*, pages 252–259, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-026-4.

Mark Stemm and Randy H. Katz. Measuring and Reducing Energy Consumption of Network Interfaces in Hand-held Devices. In *IEICE Transactions on Communications, vol. E80-B, no. 8*, pages 1125–1131, 1997.

STMicroelectronics. Inertial Sensor: Low Voltage 3Axis - 2g/6g Digital Output Linear Accelerometer. Part number LIS3LV02DQ. Technical datasheet. Revision 1, 2005.

Kerri Stone and Michael Colagrosso. Efficient Duty Cycling through Prediction and Sampling in Wireless Sensor Networks. *Wireless Communication and Mobile Computing*, 7(9):1087–1102, 2007. ISSN 1530-8669.

Jing Su, Kelvin K. W. Chan, Andrew G. Miklas, Kenneth Po, Ali Akhavan, Stefan Saroiu, Eyal de Lara, and Ashvin Goel. A Preliminary Investigation of Worm Infections in a Bluetooth Environment. In *WORM'06: Proceedings of the 4th ACM Workshop on Recurring Malcode*, pages 9–16, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-551-9.

Yanjun Sun, Omer Gurewitz, and David B. Johnson. RI-MAC: a Receiver-initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks. In *SenSys'08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 1–14, New York, NY, USA, 2008. ACM Press. ISBN 978-1-59593-990-6.

Suresh Singh and Cauligi Raghavendra. PAMAS: Power Aware Multi-Access Protocol with Signalling for Ad Hoc Networks. *ACM Computer Communications Review*, 1999.

Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998. ISBN 978-0262193986.

TexasInstruments. CC2420 Single-chip 2.4 GHz IEEE 802.15.4 Compliant and ZigBee-Ready RF Transceiver (Rev. B). Product Datasheet, Revision SWRS041b, 2008.

D. Thompson, P. S. Hammond, K. S. Nicholas, and Michael A. Fedak. Movements, diving and foraging behaviour of grey seals (Halichoerus grypus. *Journal of Zoology*, 224:pp. 223–232, 1991.

Steven K. Thompson and George A. F. Seber. *Adaptive Sampling*. Wiley-Interscience; 1st Edition, May 1996. ISBN 978-0471558712.

Amin Vahdat and David Becker. Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-200006, Duke University, April 2000.

Yong Wang, Margaret Martonosi, and Li-Shiuan Peh. A New Scheme on Link Quality Prediction and its Applications to Metric-based Routing. In *SenSys'05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 288–289, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-054-X.

Yong Wang, Margaret Martonosi, and Li-Shiuan Peh. A Supervised Learning Approach for Routing Optimizations in Wireless Sensor Networks. In *REALMAN'06: Proceedings of the second International Workshop on Multi-hop Ad Hoc Networks: From Theory to Reality*, pages 79–86, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-360-3.

Richard Weber. The Optimal Strategy for Symmetric Rendezvous Search on K3. *ArXiv e-prints: 0906.5447 [math]*, June 2009.

Rebecca Willett, Aline Martin, and Robert Nowak. Backcasting, Adaptive Sampling for Sensor Networks. In *IPSN'04, Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pages 124–133, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-846-6.

Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed Energy Conservation for Ad Hoc Routing. In *MobiCom'01: Proceedings of the 7th Annual International Conference* *on Mobile Computing and Networking*, pages 70–84, New York, NY, USA, 2001. ACM Press. ISBN 1-58113-422-3.

Wei Ye, John Heidemann, and Deborah Estrin. Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, 2004. ISSN 1063-6692.

Wei Ye, Fabio Silva, and John Heidemann. Ultra-low Duty Cycle MAC with Scheduled Channel Polling. In *SenSys'06: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 321–334, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-343-3.

Rong Zheng, Jennifer C. Hou, and Lui Sha. Asynchronous Wakeup for Ad Hoc Networks. In *MobiHoc'03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 35–45, New York, NY, USA, 2003. ACM Press. ISBN 1-58113-684-6.

Rong Zheng, Jennifer C. Hou, and Lui Sha. Optimal Block Design for Asynchronous Wake-Up Schedules and Its Applications in Multihop Wireless Networks. *IEEE Transactions on Mobile Computing*, 5(9):1228–1241, 2006. ISSN 1536-1233.