Scalable Data Collection for Mobile Wireless Sensor Networks

Navid Hassanzadeh

KTH Electrical Engineering

Degree project in
Electrical Engineering
Master of Science,
Stockholm, Sweden 2012

XR-EE-LCN 2012:001

KTH Electrical Engineering

Master of Science Thesis

Scalable Data Collection for Mobile Wireless Sensor Networks

Navid Hassanzadeh

Supervisors:

Dr. Thiemo Voigt (SICS)

Dr. Olaf Landsiedel (KTH-Laboratory of Automatic Control)

Examiner:

Professor Gunnar Karlsson (KTH- Laboratory of Communication Networks)

Swedish Institute of Computer Science

School of Electrical Engineering
Kungliga Tekniska Högskolan
Stockholm, Sweden

Swedish Institute of Computer Science
Stockholm, Sweden

Stockholm, Sweden – 30 November 2011

Abstract

In the near future WSNs (wireless sensor networks) which consist of tiny wireless embedded systems will be an inseparable part of our daily lives. Data collection, collecting data from a large number of sources to one or more base stations, is a typical application for WSNs. A substantial number of data collection algorithms have been specifically designed for static scenarios while there are some scenarios in which sensor nodes are attached to intrinsically mobile objects. Generally, in such scenarios delay tolerant networking approaches have been exploited for offline data analysis. However, in-situ data collection from mobile scenarios has received little attention.

We propose Mobile Collect to address the limitations of static data collection protocols in mobile scenarios. For this purpose, Collection Tree Protocol (CTP), a de facto standard for data collection, which is implemented in Contiki-OS (Contiki Collect), has been optimized to avoid loops and to react quickly to topology changes which occur frequently in mobile scenarios. The MAC (Medium Access Control) layer in WSNs has a decisive impact on the overall performance of mobile networks in terms of power consumption, and packet delivery rate. We have evaluated Mobile Collect protocol with a receiver-initiated (A-MAC that we implemented in Contiki-OS) and a sender-initiated (Contiki-MAC) MAC protocol.

Compared to the Contiki Collect and the recently proposed DYMO (Dynamic MANET On-demand) protocol, Mobile Collect with Contiki-MAC shows a significant improvement in reliability while it has a slight increase in power consumption. A-MAC slightly improves reliability for sparse topologies, but has higher power consumption.

Acknowledgments

This thesis has been done at the Networked Embedded System group at the Swedish Institute of Computer Science. I would like to take the opportunity and express my deepest gratitude towards my supervisors Thiemo Voigt and Olaf Landsiedel. Thiemo provided me this unique opportunity to do my master's thesis in one of the prestigious research institute in wireless sensor networks field and he patiently guided me during this thesis work. Olaf advised me in all stages of the thesis from design to writing the report. During summer holiday I was only in touch with him. His guidance during this period was crucial to keep the work continuous. Additionally, I would like to thank my examiner, Professor Gunnar Karlsson, for his follow up and his invaluable advice about scientific writing. I am also thankful to all researchers in NES especially Niclas Finne for his precious help and comments. Also, I would appreciate my parents for all their supports and encouragements during last two years. I could not have completed my Master's degree without them.

This work has been performed within the SICS Center for Networked Systems funded by VINNOVA, KKS, SSF, ABB, Ericsson, Saab SDS, TeliaSonera, T2Data, Vendolocus and Peerialism. Main partner was Vendolocus Development AB and we are very thankful to Peter Stenlund for interesting discussions and encouragements.

Contents

List of Figures

Chapter 1

Introduction

Wireless sensor networks (WSNs) consist of a large number of sensor nodes, tiny wireless embedded devices with sensing, processing and communication abilities, distributed in a field to measure and monitor different physical phenomena such as temperature, pressure and motion. Moreover, sensor nodes could be attached to other objects such as animals, humans and robots to gather information about these objects or their surrounding environments. Sensor nodes are usually battery-operated devices. Sometimes due to the nature of the application, it is impossible to replace the batteries, thus energy consumption is at the top of the list of performance evaluation. Limitation in memory capacity and processing power are considered as other challenges in WSNs.
Data collection is an important application for WSNs. Data collection algorithms are designed to gather information in applications such as wild life monitoring, environmental monitoring, noise pollution monitoring and battlefield sensing. Traditional WSNs consider static scenarios where sensor nodes have fixed position. The algorithms designed for these networks only tolerate small dynamics such as joining new nodes to the network or faulty nodes that causes small changes in the original topology. However, there are a large number of applications that are intrinsically mobile which adds a new challenge to WSNs. Usually, DTN (Delay Tolerant Networking) approach has been used to tackle mobility. For example in [30], a pioneering work in this context, a sink node occasionally comes to the vicinity of a group of sensor nodes that are mobile and collects their stored data which can be used for off-line data analysis. At the opposite extreme, [27] considers an on-line data gathering for mobile sensor nodes that have a long range radio such as GPRS (General Packet Radio Service) connection to a fixed base station. However, in this application every sensor node must have the GPRS connection which is not economical. Also, in this application we do not take advantage of inter-nodes communication. In other words we do not exploit the small radio chip on the sensor nodes.

1.1 Problem Statement

In this work we consider data collection for mobile wireless sensor networks. Similar to other sensor network applications we need to take the performance of energy consumption into consideration. In contrast to the traditional wireless sensor networks which are generally stationary, we consider mobility for both types of nodes — sources and sinks. The thesis especially consider data collection from skaters in a rollerblading competition where sensor nodes are attached to the athletes to collect different statistics such as temperature, heart beat, speed, etc. Among these attached sensor nodes, only a few of them have facilities such as GPRS to connect to the infrastructure based network like GSM (Global System for Mobile Communication) . Hence, the other nodes should find a path to send their data to these sensor nodes. Obviously, the network will have a lot of transient links. Therefore, mobility is another challenge for such applications.

1.2 Method

In this thesis we started with a general study of wireless sensor networks with emphasize on routing layer (data collection) and MAC (medium access control) layer. We have tried to cover all related works that consider mobility of sensor nodes. Most of the related works do not match exactly to our requirements in this thesis. We decided to chose CTP (Collection Tree Protocol), the most well-known algorithm for data collection, as a starting point. We studied CTP in detail and its counterpart in Contiki OS, Contiki Collect, to find out its performance in mobile scenarios. Then, we proposed some solutions to overcome the drawbacks of CTP in mobile scenarios. As a next step, we evaluated an optimized CTP algorithm with different MAC protocols. Especially, we are interested in sender-initiated and receiver-initiated classes of the MAC layer. Considering receiver-initiated MAC layer, we have implemented A-MAC, the most recent receiver-initiated MAC layer, in Contiki OS. Evaluating the new algorithm, we examined its performance in comparison with Contiki Collect and TYMO which is the implementation of DYMO in TinyOS.

1.3 Limitation

This work has been conducted based on synthetic mobility models. For example, we have exploited highway mobility model to emulate the movements of skaters in a rollerblading competition. However, a better evaluation demands real mobility traces of the applications. Moreover, we should have evaluated it with different topologies to make a better judgment about the performance of the designed algorithm. Dimensioning the number of sink(s) is another challenge. To have a proper dimensioning we also need traces of real mobility.

1.4 Scientific Contribution

To our knowledge this is the first work that considered data collection from a mobile wireless sensor network where all nodes are mobile. The work is based on a well-known data collection algorithm, CTP (Contiki Collect), which is designed for stationary scenarios. Having optimized some parts of the Contiki Collect, we managed to use it in mobile scenarios. In our evaluation scenarios we achieved considerable improvement in reliability while the power consumption is almost the same. Furthermore, the performance of the new algorithm is identical to the performance of the default algorithm in static scenarios. Also, in the context of this thesis we implemented the most recent receiver-initiated MAC protocol, A-MAC, which could be considered as a contribution to Contiki-OS

1.5 Outline

Chapter 2 gives a quick overview of the internal structure of sensor nodes followed by the different aspects of sensor-node-networking. It explains some important medium access control protocols for sensor-nets. This chapter also contains a brief review of existing data collection algorithms. Chapter 3, the main chapter of the thesis, covers the detailed explanation from design to implementation of a data collection algorithm that considers the mobility of sensor nodes. Chapter 4 demonstrates conducted experiments as well as the evaluation of the new algorithm. Finally, in chapter 5 we provide a conclusion for this work and consider some possible future work.

Chapter 2

Background

In this chapter we give an explanation about the basic principles behind the thesis. First, we start with a quick overview of wireless sensor networks, building block of sensor nodes and some details about the physical layer and the radio chip. Then, we introduce the operation and the distinct features of some well-known MAC protocols. It is followed by explaining the mechanism of data collection which are an important part of this thesis. Next, a typical classification of mobility models is explained. Then, some important terminologies related to MWSN (Mobile Wireless Sensor Network) are given. Finally, some related work to the thesis will be presented.

2.1 Wireless Sensor Networks

2.1.1 Introduction

Convergence of micro-electro-mechanical systems (MEMS), embedded system and wireless communication has opened a new era in mobile networking and pervasive computing which is referred to as “wireless sensor networks”. In WSNs, sensor nodes or motes[1] are usually deployed in the order of hundreds or thousands. Motes perform sensing and perhaps limited processing. At last, wireless network of motes which is shaped on an ad hoc basis, is responsible to disseminate captured information. Currently, low production cost, tiny size and wireless communication make the embryonic ideas such as the internet of things and ambient intelligence more feasible than at any time before. It is expected that unattended and unaccompanied sensor nodes should be operational for a long time. Therefore, wireless sensor networks open a new research area which attempts to devise power efficient solutions. It is also worthwhile to mention that any solution should consider application-specific requirements. For example, in a mission-critical application having a reliable and real-time network is a necessity while power efficiency is not the first priority.

[1]A very small piece of dust (Longman Dictionary); in wireless sensor network nodes are known as motes.

The reminder of this section is organized as follows: first sensor nodes are considered as an individual entity, next we consider the role of a sensor node in a network and how a typical wireless sensor network works; finally we compare WSN with MANET (Mobile Ad hoc Network).

2.1.2 Architectures: from Motes to Network

This section is divided into two sub-sections. First, hardware construction of sensor nodes and second, networking aspects are considered.

Hardware Architecture

Abstract View. Generally, a sensor node has four main components: processor, transceiver, sensing unit and power unit [7]. Figure 2.1 shows a typical internal construction of a sensor node.

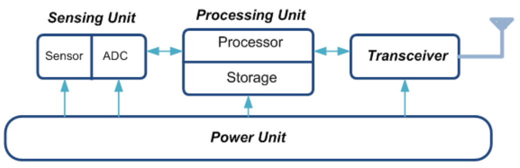


Figure 2.1: Building blocks of a typical sensor node.

The processor unit is the central unit of a mote. It is responsible for controlling and coordinating the operation of other components. It performs different tasks in the form of code that a user supplied to it. The sensing unit contains two sub-units: sensor and ADC (Analog to Digital Converter). Sensor is a device that performs sensing a phenomenon, yet output of sensor is analog signal which is not readable for the processor and thus, the ADC is needed to convert analog signal to digital form. Transceiver is the enabling component for wireless ad hoc networking of sensor nodes. A power unit is necessary to supply power to the components while it meets their specific requirements.
Hardware platform. To have a concrete mote, each building block of Figure 2.1 should be specified. For example, processing unit can be one of MSP430, ATmega128L, etc microcontrollers and for the transceiver unit it is possible to have one of the Chipcon CCxx radio chips families. Thus, different sensor modules are recognized based on their physical electronic components. These sensor node modules are well-known as hardware platform such as Micaz [1], TmoteSky [3], SunSPOT [1, 2], etc. Transceiver or the radio is the most relevant component of these platforms to this thesis work. In the next section some features of CC2420 [14], the radio chip for TmoteSky platform, has been introduced. It is important to mention that we have emulated the TmoteSky sensor nodes in the simulations.

Network Architecture

Communication Architecture. Sensor nodes have been deployed in a large scale to observe and collect data about a phenomenon. Ultimately, it is intended to send the collected data to a base station. In other words, the communication architecture in WSNs follows source to sink paradigm (convergecast) of routing algorithms. In WSNs, sources are deployed in a large number but it is possible to have single sink or multiple sinks. Figure 2.2 depicts a typical multiple sinks scenario. Interestingly, this architecture is the main motivation to develop collection algorithm which we will see in Section 2.3.

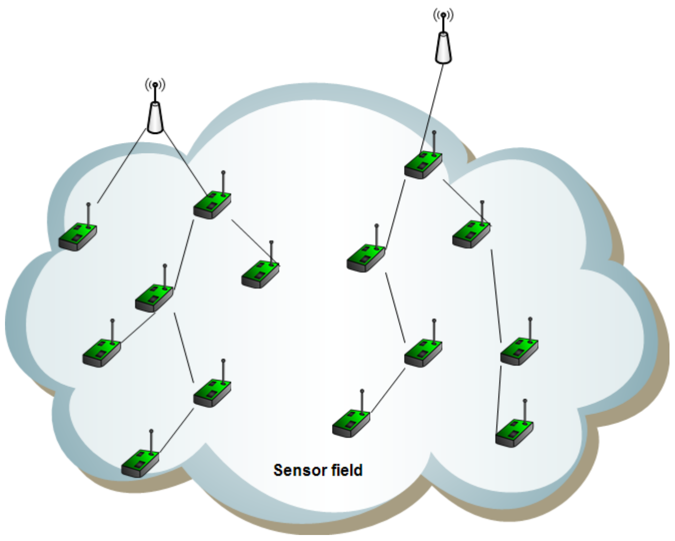


Figure 2.2: Wireless sensor network with multiple sinks.

Protocol Stack. Like other communication networks, wireless sensor networks also should be designed based on layering guidelines. Figure 2.3 illustrates abstract layers and planes in a WSN. Although it is similar to TCP/IP, protocols designed for TCP/IP models cannot accommodate properly in WSN stack. Intrinsic limitation of WSN demands a lightweight and power efficient design. For instance, using TCP protocol in wireless network especially in WSN is not easily achievable. Furthermore, addressing constraints in WSN, usually protocol designers exploit cross layer optimization in which layers can talk not only to their immediate layers above and below ; also, they can break layering boundaries to communicate with the other layers.

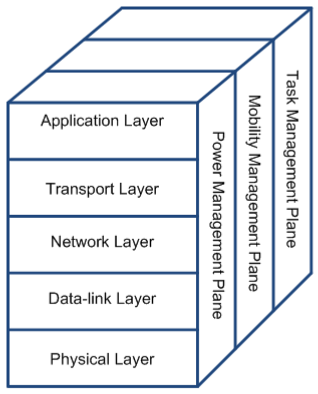


Figure 2.3: General protocol stack for WSN [7].

Three planes in Figure 2.3 are new compared to TCP/IP protocol stack. These planes are added to address unique restrictions in WSN. Power management plane deals with power limitation in sensor nodes. For example, turning the radio off when there is no transmission or reception. Mobility management plane manages the possible motion of sensor nodes, updates neighbors table and detects new neighbors. Task management plane is responsible to schedule tasks among sensor nodes. For instance, imagine that some sensor nodes are located in a small region of a large sensor field and that they are responsible to measure temperature in that region, clearly not all of the sensors are needed to do this task. Based on some criteria such as amount of power, task management plane specifies which sensor node(s) must do the measurement task. In the following sections data link and network layers are discussed. In the last part of this section some details of physical layer which is relevant to the thesis will be introduced.

Physical Layers. Sensor nodes widely make use of IEEE 802.15.4 [5] as physical layer. CC2420 used in TmoteSky, is a radio chip that implements IEEE 802.15.4 core functionalities which can be summarized as follows [5]:

- Activation and deactivation of radio transceiver.
- Energy detection (ED) within current channel.
- Link quality indicator (LQI) for current packet.
- Clear channel assessment (CCA) for carrier sense multiple access with collision avoidance (CSMA/CA).
- Channel frequency selection.
- Data transmission and reception.
- Transmission power control (CC2420 support programmable output in 8 steps from approximately –24 to 0 dBm)

Three frequency bands are supported by IEEE 802.15.4: 2450 MHz band with 16 channels, 915 MHz band with 10 channels and 868 MHz band with 1 channel. Frequency of sixteen channels for 2450 MHz band, which we are interested in, are calculated as follows:

$$F_c = 2405 + 5(k - 11)\ MHz \qquad for\ k = 11, 12, 13, \ldots, 26.$$

It should be noticed that $k = 0$ and $k = [1, 10]$ are reserved for 868 MHz and 915 MHz bands respectively.

2.1.3 MANET vs WSN

Generally, all applications of wireless sensor networks contain sensing, possibly doing a limited process and finally collecting data to a specific node known as a sink. Therefore, routing protocols used for collecting data have an important effect on the performance of the system. At first, it seems that it is possible to use protocols that are developed in MANET, yet, WSN and ad-hoc networks have different characteristic. They are varying as follows [7]:

1. The number of sensors in WSN can be much higher compared to that of nodes in MANET.
2. Nodes' lifetime consideration has decisive influence on protocol design while in MANET energy issue is not the main concern.
3. Collecting data in a sink(s) is a promising goal in WSN, while in MANET, there is no source to sink communication.

It is easy to see that using MANET protocols in WSN is not a trivial matter and each of them has their own requirements in the protocol design process.

2.2 MAC Layer in WSN

Wireless channel has a broadcast nature and, in turn, sensor nodes share the wireless channel when they are within the transmission range of one another. Obviously, it is essential for nodes to have collision-free access to the wireless channel. The MAC layer manages wireless channel access. Traditionally, MAC layer concentrates on delay and throughput as the performance metrics. However, energy constraints in sensor nodes enforce energy-aware MAC layer design as a top priority for wireless sensor networks in which battery-operated sensor nodes are presumed to work for a long time. Moreover, protocol design such as MAC protocols for wireless sensor networks highly depend on the intended application. That is why it is not possible to have a standard MAC layer and we can see a large number of MAC protocols having emerged.

2.2.1 Origins of Energy Consumption

Generally, processing and communication are two major sources of energy consumption. Communication is the main factor [7] in energy consumption and it can be decomposed as follows [18]:

Idle Listening: in this case radio is "on" and listens to the wireless channel without receiving any packet. Turning off the radio is an efficient way to avert energy consumption.

Collision: when two or more sensor nodes transmit a packet to the same receiver at the same time, collision happens. To compensate this defect in transmission, sender retransmits collided packets, so collision wastes energy on retransmission.

Overhearing: this means that a sensor node receives a packet that is targeted to another node.

Over-emitting: Transmission of a packet when its destination is sleeping and is not able to receive. This can be considered as an unsuccessful transmission that results in waste of energy.

Control Overhead: Control overhead increases the packet size and, in turn, a large packet needs more transmission time and consumes more energy.

2.2.2 MAC Protocols

Generally, MAC protocols are divided in three major classes: reservation-based, contention-based and hybrid. This classification is based on the medium access techniques. The reservation-based uses time division multiple access (TDMA) by which each sensor node has access to the medium in predefined time slots. The contention-based approach utilizes CSMA (Carrier Sense Multiple Access) algorithm for the medium access and the hybrid method is a combination of contention and reservation methods. Z-MAC [40] could be one example of hybrid class. Contention-based approach can be further classified as synchronous such as S-MAC [49] and asynchronous such as B-MAC [39]. In synchronous protocols, sleep /wake up times are synchronous among sensor nodes or a cluster of sensor nodes, but in a asynchronous algorithm each node has independent duty cycle. Recently, some MAC protocols developed (RI-MAC [48], LPP [38] and A-MAC [22]) with which it is possible to classify MAC-protocols in two categories: Receiver-initiated and Sender-initiated. In receiver-initiated MAC, receiver sends probe packets to declare that it is ready to accept a pending packet of a sender. In other words, a receiver node actively takes part in communication while in sender-initiated paradigm, a sender node sends data when it has any. In the next section some well-known MAC protocols are introduced.

B-MAC

B-MAC protocol described in [39] is based on two mechanisms: low power listening (LPL) and clear channel assessment (CCA). Also it provides an interface for upper layers to adjust CSMA parameters which facilitates cross layer design.

Low Power Listening. Fixed duty cycle is one of the drawbacks in S-MAC. LPL (preamble sampling) is intended to get rid of the synchronous schedule for sensor nodes. In other words, with LPL each node can determine its own sleep and wake up time. The main idea in LPL is to send a preamble before sending the actual packet. Therefore, each node (receiver) wakes up for a small period of time. If it detects a preamble, it will remain awake to get the subsequent packet; otherwise it goes back to sleep. Channel listening interval is a vital parameter for efficient energy management and it is determined by traffic load.

Clear Channel Assessment. It is a mechanism to detect channel activity. Ambient noise fluctuation leads to inaccurate operation such as positive false[2] which means waking up when there is no activity in the channel. To improve busy channel detection process, CCA mechanism is done in two phases: noise floor estimation and signal detection. CCA estimates the noise floor after sending data (idle channel) and the noise floor estimation is calculated by means of an exponential moving average of samples. In the signal detection phase, before sending a packet, CCA takes some samples and checks them for outliers. Lack of outlier means a stable signal strenght, in other words there is an ongoing tranmission in the channel.

As mentioned before, unlike S-MAC, B-MAC uses an asynchronous approach where each node decides about its listen and sleep periods. B-MAC uses a long or extended preamble which must be greater than the maximum receiver sleep period. This long preamble poses some drawbacks as follows:

1. Receiver has to wait until preamble is finished then it is possible to exchange data. In the worst case the receiver has woken up at the same time that the preamble starts.

2. Overhearing is another downside in low power listening. It happens when a non-target node wakes up due to existence of preamble and at the end of preamble it finds out that it is not the destination for the current transmission.

3. Target node has to wait for the whole duration of preamble before receiving data. As a result, each hop has delay equal to at least one preamble length. Specially, in multihop communication accumulated delay has considerable impact on the network delay performance.

[2]Positive false happens as a result of the approach of the threshold where channel samples are compared with noise level to determine free channel

X-MAC

As mentioned in the previous section, extended preamble in B-MAC is prone to waste energy. X-MAC [11] proposes shortened preamble to compensate drawbacks of B-MAC. Shortened preamble is described as follows:

1. X-MAC divides long preamble into series of short preamble packets each of which contains the address of destination, this prevents overhearing problem and non-target nodes can quickly go back to sleep.

2. X-MAC inserts a small delay between two consequent short preamble packets. This provides opportunity for the target node to send an immediate acknowledgment to declare its readiness for receiving data packets; as a result the sender cuts off the preamble phase. This mechanism is called strobed preamble which improves the latency and the performance of energy consumption. Additionally, strobing mechanism is useful when there are some transmitters wishing to send data to a receiver. Every transmitter sends its own preamble in the hope of waking up the receiver, but the receiver has already woken up by one of them and the others are not anymore needed to transmit preamble. Figure 2.4 shows strobing mechanism in which a receiver sends an early acknowledgment by which a transmitter is informed that the receiver is awake and stops sending short preambles.

3. X-MAC also introduces adaptive algorithm to dynamically adjust the duty cycle based on the traffic load of a network. For example in a tree topology nodes closer to a sink deal with large amount of data. Adaptive duty cycling is designed to achieve optimal performance rather than fixed duty cycle which is suboptimal.

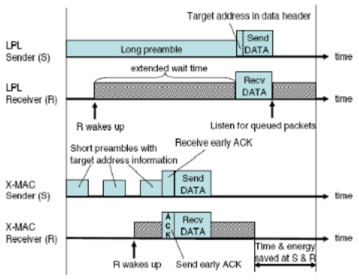


Figure 2.4: Extended preamble vs. short preamble [11].

Contiki-MAC

Contiki-MAC, the default MAC protocol of Contiki OS [20], has been introduced in [21]. In fact it is an amalgam of other MAC protocols. Specifically, it is based on low power listening of B-MAC [39], packetized preamble of X-MAC [11], using data packet as a wake-up signal in BoX-MAC [37] and the phase lock mechanism of Wise-MAC [24]. Figure 2.5 shows a typical operation of Contiki-MAC.

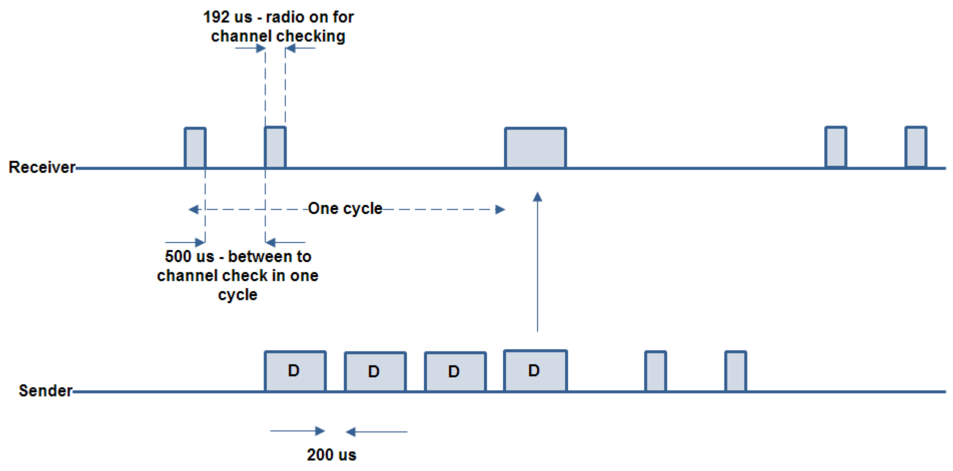

192 us - radio on for channel checking
Receiver
One cycle
500 us - between to channel check in one cycle
Sender
D
D
D
D
200 us

Figure 2.5: Basic operation of Contiki-MAC.

During one cycle Contiki-MAC listens to the channel twice a cycle, with a gap of $500\mu s$ between two listen time and each listen time lasts $192\mu s$. A sender node sends a train of identical data packets. A train of data packets is similar to sending preamble which is a technique to wake up the receiver. There is an optional phase optimization in Contiki-MAC with which, similar to Wise-MAC, a sender learns when the intended receiver will turn its radio on to listen to the channel.

Low Power Probing (LPP)

The idea of low power probing is introduced in [38]. It is used to provide a receiver-initiated mechanism for data collection (Koala algorithm) where LPP used as a technique to wake up sender nodes in a network. In fact, LPP is an asynchronous and power-saving MAC protocol with which a receiver periodically broadcasts small packets called probes to declare that it has been woken up and is ready to receive possible data packets. After probing, the receiver keeps its radio "on" for a while to provide a chance for a potential sender to send acknowledgment and to declare that it has data to send. Figure 2.6 shows a typical LPP operation.

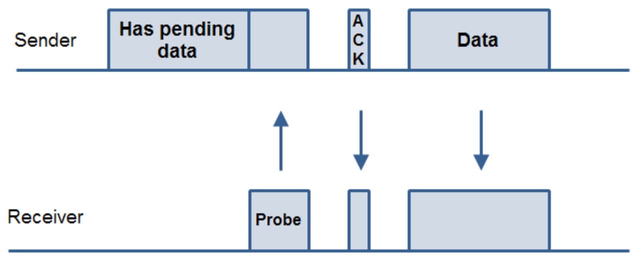


Figure 2.6: Basic operation of LPP.

2.3 Data Collection in WSN

Environmental and habitat monitoring are applications where sensor networks have been widely used. In such applications sensor nodes send their data to sink node(s). The life time of the network is the major concern, mainly because in these applications sensor nodes span a large area and it is not easy to replace dead-batteries in nodes. Even though, in some applications like animal monitoring [40] or athletes monitoring in an interminable competition like rollerblading or Marathon, it is inevitable to leave network unattended while it is kept up and run properly. Data collection protocols are routing protocols designed to collect or gather data as power efficient as possible from sensor nodes to sink nodes. The rest of this section discusses about data delivery models and some novel data collection protocols.

2.3.1 Data Delivery Models

Data delivery to a sink can be classified as continuous, event driven, query-driven and hybrid [6]. In continuous model each sensor node periodically sends its data to a sink. In an event-driven model, transmission of data starts when an event happens. In query based model, a sink sends query and asks for data and finally hybrid model is combination of previously mentioned models. Data delivery model is influential factor in choosing an appropriate routing protocol.

2.3.2 Data Collection Protocols

As mentioned before, collection protocols are used to build and maintain a routing tree in which a sink is the root of the tree. Basically, there are three major design considerations in collection protocol [41]: link dynamics, transient loops and duplicate packets. Link dynamics are due to the natural characteristic of wireless link where the coherence time[3] is in the order of a few hundred milliseconds [26]. Transient loops are similar to the loops in other distance vector algorithm such as RIP. Duplicate packets are the outcome of lost acknowledgments. Moreover, in a scenario with changing topologies, specifically in mobile scenarios, these challenges become more severe. This section provides an overview of CTP (Collection Tree Protocol) [26] and its implementation in Contiki OS (Contiki Collect).

Collection Tree Protocol

Collection Tree Protocol (CTP) [26, 42] is a state-of-the-art protocol designed for data collection in wireless sensor networks. Its specification is provided in TEP 123 [42] of TinyOS. Link quality metric which is used to select potential parent is one of the distinguishing features among different collection tree algorithms. CTP uses ETX (expected transmission) as a routing metric which is first introduced in [15]. It reflects the number of transmissions, including

[3]Coherence time is a time duration that channel impulse response is not changing.

retransmissions that are needed to deliver a packet without error to a destination. For example, assume that it is needed to have two transmissions to deliver a packet to a destination, then the ETX is equal to 2. Clearly, a perfect link has ETX equal to 1. On the one hand, a link estimator is responsible to calculate ETX for neighbors. This ETX is called one hop ETX (ETX_{1hop}). On the other hand, neighbors advertise their own accumulated ETX (sum of all ETX in the path from neighbor to root) which is called multihop ETX ($ETX_{multihop}$). As a result, each node can estimate cost (ETX) for each route ($ETX_{multihop}$ + ETX_{1hop}). Accordingly, the lowest cost gives the desirable route. It is worth mentioning that for sink node ETX is equal to zero. CTP has the following components [42]:

- ***Routing Engine (RE):*** This component is responsible for sending and receiving beacons as well as creating and updating the routing table. Routing table contains the list of neighbors from which it is possible to select the parent. As mentioned before ETX is used as the parent selection metric.

- ***Forwarding Engine (FE):*** On the one hand, when packets come from the application layer, FE is responsible to deliver it to the lower layer. On the other hand, when packet comes from the MAC layer, FE is responsible to send it to the application layer. Detecting and repairing routing inconsistencies (loops) as well as handling duplicate packets are other vital tasks of FE.

- ***Link Estimator (LE):*** One hop link estimation (ETX_{1hop}) is performed by LE. This estimation is based on the statistics that are gathered from beacons over time and successful transmission for both inbound and outbound links. (ETX is estimated by both incoming and outgoing links).

[26] introduces two innovations in CTP: data path validation and adaptive beaconing.

- ***Data path validation:*** The link estimator relies on the beacon packets which means the link estimator can only capture topology change in the order of the beacon interval, in other words this means reaction to the dynamics of the network is slow. Data path validation suggests including routing information in data packet as well as beacon packets. Therefore, network responses to the changes in first data packet. For instance if a node receives a data packet with the ETX (routing information) less than its own ETX, it can be concluded that there is a strong probability of a loop occurrence.

- ***Adaptive beaconing:*** Beacon packets are main means to have an updated routing table. However, a fixed interval beaconing poses a tradeoff between energy efficiency and an agile routing table update. For large beacon intervals, the network might suffer from routing loops as the consequence of obsolete routing table. On the contrary, frequent beacon packets is not energy efficient. Addressing this problem, CTP introduces adaptive beaconing which is based on the Trickle algorithm [34]. To put it simply, CTP starts with initial interval of 64 ms and doubles it until an interval of one hour, but, if one of the following events happens, routing layer resets the interval value to the initial value.

1. Routing layer is asked to send packets to a node whose ETX is higher than its own ETX (possibility of loop).
2. Current node has a new route that is significantly less costly than the previous one and nearby nodes should be informed about this new route (significant is ETX of 1.5).
3. If a node receives a packet with P (polling) bit set which shows a request by newly joined node that needs a routing update.

Contiki Collect

Contiki Collect is a CTP-like collection protocol integrated in Contiki OS. Similar to the CTP it is based on ETX as a route selection metric. Figure 2.7 shows Contiki Collect in the protocol stack of Contiki OS. RDC (Radio Duty Cycling) is a particular layer in Contiki OS that should be clarified. In fact, MAC protocols for WSNs are different from each other in the way that they handle turning on/off the radio for efficient power consumption while CSMA (traditional MAC) protocol is not directly associated to the duty cycling and usually it is unchanged. Thus, adding RDC simplifies design process by separating CSMA from duty cycling.
As can be seen from Figure 2.7, Contiki Collect contains four building blocks. The role of each block is described as follows.

- ***Link Estimator:*** This block is closely related to the ETX_{1hop} calculation. It stores estimation history (8 estimation is default value). Based on these values ETX_{1hop} is calculated. Furthermore, link estimator updates ETX_{1hop} according to the number of (re)transmission needed for a successful packet delivery. As a reminder, it should be mentioned that low link quality (failure and retransmission) leads to increase in the ETX_{1hop} value.

- ***Neighbors Manager:*** This block is responsible to keep information about detected neighbors or in networking terminology it provides the routing table for a sensor node. Each entry (route) in the routing table contains address (node ID) of a neighbor and its advertised ETX (rtmetric[4]). Moreover, it contains two timers: one timer to remove an entry which is not updated for a long time and another timer to remove link estimation information in link estimator block. Moreover, an algorithm for parent selection is implemented in the block of neighbor management.

- ***Neighbor discovery:*** As the name says, this block is responsible for detecting nodes in the communication range of the current node. It broadcasts neighbor discovery packets

[4]In Contiki Collect, “ETX” referred to as ETX_{1hop} and “rtmetric” referred to as $ETX_{multihop}$

(beacon) in which “node ID” and “rtmetric” are included. It uses an adaptive approach to broadcast beacons. However, for the quick propagation of routing information, Collect block (see below) could force neighbor discovery to restart its beacon timing.

- ***Collect:*** This block is the central unit for the collect algorithm. It provides interfaces to the upper and lower layers and initializes other previously mentioned modules in Contiki Collect. It handles sending and receiving packets as well as their acknowledgments. If there is a major change in current node, it forces the module of neighbor discovery to broadcast a beacon to inform other nodes in the neighborhood about recent reorganization.

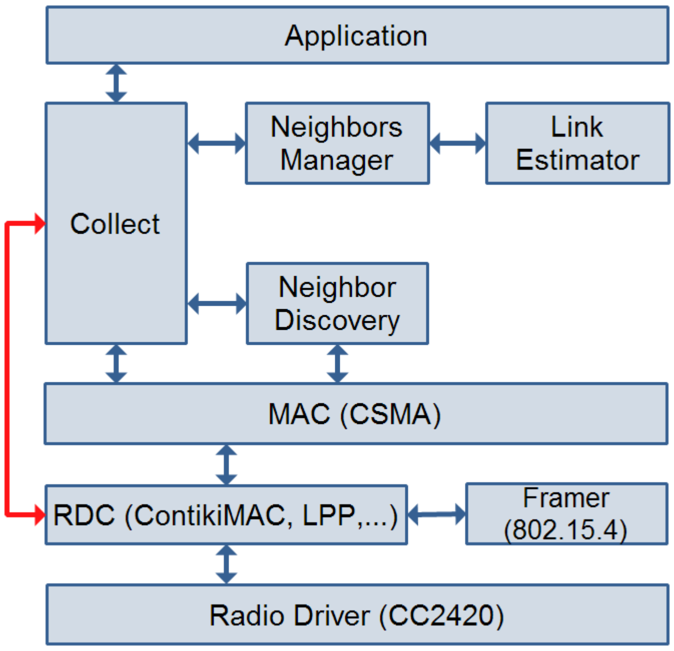


Figure 2.7: Communication stack of Contiki OS. The interface between Collect and RDC (Radio duty cycling) will be discussed in the design chapter.

2.4 Mobility Models

Mobility is a distinguishing characteristic of MANET, VANET (Vehicular Ad-Hoc Network) and MWSN from static networks. Mobility introduces new challenges to the protocol design and has crucial effect on the network performance. Thus, it is essential to model mobility by which it is possible to capture change in location, velocity and acceleration of mobile nodes over time. Moreover, traffic pattern is another issue that should be carefully considered for protocol design. This section gives an overview of some mobility models.
Basically, a mobility model is developed based on different scenarios that we encounter in our daily life. For example a highway model attempts to emulate mobility of vehicles in a highway. However, in some scenarios such as mobility of students in a campus it is not trivial to have

analytical model, in such cases mobility traces database such as [16] provides accurate and reliable information.

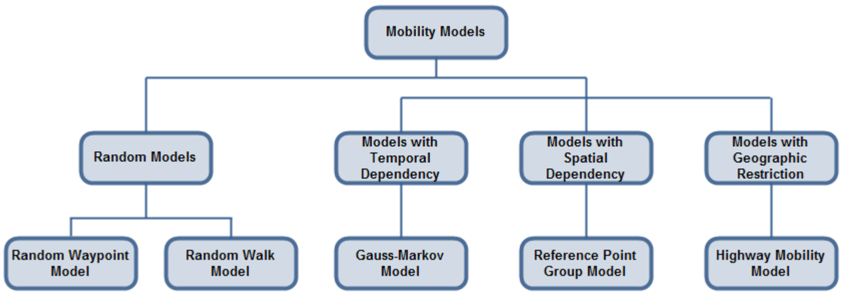

Mobility Models
Random Models
Models with Temporal Dependency
Models with Spatial Dependency
Models with Geographic Restriction
Random Waypoint Model
Random Walk Model
Gauss-Markov Model
Reference Point Group Model
Highway Mobility Model

Figure 2.8: Simple classification of mobility models.

Figure 2.8 shows a concise classification of mobility models. In this taxonomy, mobility models fall into four major classes. In random models, the mobility parameters such as velocity, destination, direction, etc. are chosen randomly without considering previous values. In mobility models with temporal dependency, a node's movement is influenced by the previous history of movement. In mobility models with spatial dependency, the movement of nodes correlates with each other. Finally, in mobility models with geographic restrictions, node's movement is limited to a bounded space such as street, highway, etc. Sample mobility models for each category are discussed as follows [9].

- **Random Waypoint Model:** In this model a node randomly selects its destination and a random value for its velocity from $(0, V_{max}]$. It is important to notice that velocity and destination of nodes are independent from each other. Then, nodes start traveling to the selected destination. Upon reaching the destination, nodes stops for T_{pause}. When pause time expired, nodes select new velocity and destination. On the one hand larger values for T_{pause} mean network has almost consistent topology, on the other hand small values lead to a highly dynamic network.

- **Random Walk Model:** This model has its root in physics where random walk or Brownian motion has been proposed to study unpredictable movements of particles. This model is very similar to the previous one. In fact, random walk is a variant of random waypoint. Specifically, if in random waypoint, T_{pause} is set to zero, the result is similar to the random walk model [9].

- **Gauss-Markov Model:** In this model velocity of nodes follows Gauss-Markov stochastic process in which velocity of one node is correlated over time. That means current velocity depends on the history of previously selected velocities. The degree of dependency can be determined by the randomness parameter α. If $\alpha = 0$ then Gauss-Markov model will be memoryless. Intuitively, it means Gauss-Markov model is converted to random walk model. At the other extreme, if $\alpha = 1$ Gauss-Markov model has strong memory and velocity is fixed over time.

- **Reference Point Group Mobility Model:** In RPGM all nodes are divided in groups and each group has its own leader. The movement of a group leader determines the movement of the entire group. The movement of a leader defined by a velocity vector, $\vec{Vt}_{Group}$, then every member in a group can chose its own vector with some random degree of variation from a group leader. A group of rescue teams members who follow their leader in disaster area or a group of soldiers who follow their commander in the battle field are examples of RPGM mobility.

- **Highway Mobility Model:** In highway model nodes are restricted to the lane in the highway. However, similar to the random waypoint model, nodes have certain degree of randomness (pseudo-random). For instance, randomness in velocity but the direction of motion is determined by the lane's boundary. Figure 2.9 shows highway lanes where nodes are restricted to move along them.

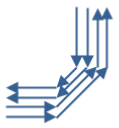


Figure 2.9: Highway mobility model where nodes are restricted in lanes.

Most network simulators support the mobility of nodes. Additionally, mobile scenario generator tools have been developed to feed mobile trace to the simulators. They are able to generate mobility pattern based on mobility models. Bonnmotion [8] is an example of such tools and it is used in this thesis work.

2.5 Mobile Wireless Sensor Networks

Recently sensor network attached to the intrinsically moving objects such as animals [30], athletes [23], vehicles [25]. This has led a new research field to emerge, named mobile wireless sensor network. MWSN, inherited all constraints introduced by static wireless sensor networks: limited bandwidth, computing ability and energy resource as the most important among all of them. Plus, mobility introduces new challenges such as inconsistent topologies, network partitioning, the need for buffer space to save data when a node is disconnected and overhead due to frequent routing table updates. Here, some important definitions related to MWSN are introduced [19].

- ***Contact time:*** It is defined as the amount of time that two nodes are in contact

- ***Contact region:*** It is a region in which there is a possibility for contact, in other words contact region is the same as communication range of a node (circle area in Figure 2.10).

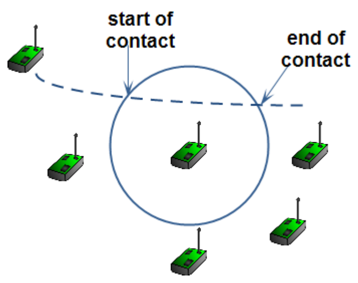


Figure 2.10: Contact region and time in mobile wireless sensor network.

- ***Residual time:*** It is a time that is used for data transmission and it is less than the contact time.

- ***Discovery:*** It is the process by which a node can spot a contact. An efficient discovery should detect as many contacts as possible while meeting the energy constraint of sensor nodes. Discovery process is divided in two groups: mobility-independent and mobility-aware discovery. In mobility aware protocols, mobility pattern exploited to estimate the probable encounter. This approach is energy efficient since nodes wake up when there is a probability to have a contact. Mobility-independent protocols are divided into scheduled rendezvous and asynchronous. In scheduled rendezvous, nodes (usually nodes and sink) agrees on the time that they can meet each other, this approach is used in Zebra Net [30] and scheduling to wake up is achieved by means of GPS. In contrast to the scheduled rendezvous, asynchronous approach relies on the duty cycling (wake up /sleep) schema of nodes which do not have agreement on their active time. Sending periodic discovery message ("beacon" or "hello") falls into this category.

- ***Types of mobility:*** Before investigating impact of mobility on a network, different kinds of mobility are discussed. Mobility is divided in two major classes: controlled mobility [44] and uncontrolled mobility which is further classified as deterministic or predictable mobility [13] and random mobility [43]. In deterministic mobility, nodes have regular or periodic contacts while in random one nodes have random contacts which can be characterized by a probability distribution such as Poisson. In controlled mobility nodes have control over their speed and trajectory, hence neighbor discovery can be handled simply in contrast to the uncontrolled mobility, moreover contact time is controllable and nodes can have contacts as much as they need.

As mentioned before mobility has a great impact on the network topology. Usually mobility is the origin of making a topology sparse or intermittent. In this case opportunistic or delay tolerant approaches are desirable such as [10, 35, 45]. However in dense and connected network where there is a path between two nodes it is possible to accommodate a traditional collection or routing protocol in mobile scenarios. Using multiple sink nodes is a handy way to have a connected network in mobile scenarios.

2.6 Related Works

Generally, most of the work in the context of mobile wireless sensor network relies on the mixed scenarios where some nodes are fixed and some others are mobile. Specifically, either sink(s) or sources are mobile. Experimentally deployed sensor networks that consider the mobility for all nodes are discussed in the context of DTN approach and they are suitable for off-line data analysis. We classify related work from different points of view as follows.

2.6.1 General Data Collection Algorithms

Up to now we have discussed CTP and its counterpart in Contiki OS (Contiki Collect) in detail. We are going to use this algorithm in the next chapter. However there are other data collection algorithms that are designed originally for stationary scenarios.

Dozer: This algorithm is introduced in [12]. It is designed to fulfill the general requirements of environmental monitoring (data gathering) systems. Network life time and reliable data transmission are the main concerns whereas latency and bandwidth have a low priority. Dozer constructs a tree topology over the physical network and guarantees that data can traverse through a loop free path to a sink node. To transfer data, it utilizes TDMA. Interestingly enough, the proposed TDMA algorithm makes use of local synchronization (one hop) rather than the global synchronization as in the traditional TDMA. In an unusual way, Dozer does not exploit MAC protocols, in fact it does not try to prevent collision. Dozer is implemented on TinyOS and practically deployed on a testbed where it achieves 0.2% of duty cycle.

BCP (Backpressure Collection Protocol): It is introduced in [36]. In contrast to the other collection algorithms it does not make end to end path. Instead of that, it routes hop by hop for each packet. Simply, the main idea in BCP is based on a queue backlog gradient which means that a packet from a node with a large backlog flows to a low backlog node. It is assumed that a sink has zero backlog. This algorithm consider the mobility of nodes. However it is designed for high data rate and it does not show a good performance in low duty cycles.

Sidewinder: Perhaps sidewinder [31] is the closest work to this thesis work. As we have mentioned before, most of the deployment of mobile wireless sensor network exploited DTN as a solution for data collection. However, this approach cannot be used for the applications that requires on-line data analysis. This work is specially interested in the inundation prediction (see below) which highly depends on the simulation and analyzing collected data. Thus it is crucial to collect flooding information without delay. In the inundation prediction application sensors nodes are placed in buoys which are floating on the water and they are equipped with GPS to send position and speed information to the mobile sink nodes. Based on the distributed knowledge of mobile sink locations, routing paths are updated. In other words, sidewinder predicts the location of mobile sinks.

2.6.2 Data Collection with Mobile Sink

There are considerable number of papers and literature that take into account mobility of sink(s) while assuming that other nodes (sources) are static. For example mobile sinks have been used to collect data from remote sensor nodes. In this case, sink mobility should be conceptually considered as a delay tolerant networking. Basically, in these algorithms mobility is considered as a merit rather than being a challenge. Some works in this filed are as follows.

Data Mule: Data mule is introduced in [43], a pioneering work in sink mobility, in which mobile agents collect and save data from sparse wireless sensor networks, then if these mobile agents are in proximity of a base station they can opportunistically forward collected data to the base station. Also, authors in [29] present advanced work in this context which uses multiple sinks to collect data. It also has load balancing algorithm for sinks.

Data Salmon: Previous work consider random movement of mobile sink(s). Although it is energy efficient but it suffers from high latency. ”Data Salmon” [17] tries to address spatial-temporal nature of data generation in environmental monitoring and surveillance applications. This characteristic of data generation causes high data rate (burst) in some part of the field that sensor nodes are deployed. In contrast to the previous work, “Data Salmon” forwards data to mobile sinks to avoid delay, but to alleviate energy consumption due to multihop routing, sinks relocate to the region with high data rate and try to decrease the number of hops.

Whirlpool: This collection algorithm is introduced in [33] and it is basically evolved from CTP. Whirlpool consider sink mobility in the network. When a sink moves, the existing distance vector tree searches the old location to find a possible neighbor node with connection to the sink node; then it quickly switches to this neighbor which has a path to the sink node. Since this work does not include any energy performance evaluation, it seems it does not consider duty cycled radio.

2.6.3 DTN Approach

Most of the well-known deployed applications that are inherently mobile take advantage of delay tolerant networking approach. Generally, DTN approach is used for the off-line data analysis. Some of works in this field are as follows:

ZebraNet: ZebraNet [30] is an application of WSN aimed to monitor wild life (zebras). Sensor nodes are attached to a group of zebras that are moving together. These sensor nodes have capability to communicate with each other and every node has the same information as others hence, communication among nodes is only to provide redundant copy of information for all nodes. To collect data, a mobile sink node (usually attached to a vehicles) comes to the vicinity of the zebras and collect their data for off-line data analysis.

BikeNet: BikeNet [23] is a deployed mobile sensing system which is used to collect data that quantifies various aspect of the cycling experience. It works in both opportunistic (default mode) and real time modes. When used in delay tolerant mode, cyclists go on a trip and upon returning home they can upload collected data. To support real time mode cyclists should have a cell phone which is used as a mobile sensor gateway.

Chapter 3

Design and Implementation

Primarily, Contiki Collect is designed for static scenarios where it provides a promising performance. This chapter begins giving an overview of the design and implementation part of the thesis. From algorithm design and implementation point of view we study Contiki Collect algorithm in mobile scenarios where it does not function properly. There are two options to have a mobility aware Collect protocol: 1) redesigning and reimplementing new algorithm. 2) optimizing Contiki Collect algorithm. To prevent redundant work we choose the second option. Thus, we should find potential sources of inappropriate operation of Contiki Collect in mobile scenarios. Then, we present some solutions to accommodate Contiki Collect in mobile scenarios. Next, we exploit solutions of A-MAC [22] to improve poor power consumption performance of LPP (the only receiver-initiated MAC in Contiki OS) to make it readily available for being used in Contiki Collect. At last, we introduce the tools that we have developed for test and verification purposes.

3.1 Overview of The Design and Implementation Phase

Figure 3.1 demonstrates a general view of this thesis work. There are two different categories in the design and implementation phase. First, we design and implement the required algorithms in Contiki OS (different layers of protocol stack). Second, it is needed to supply essential tools such as mobile scenario generator and log file analyzer for test and simulation purposes.

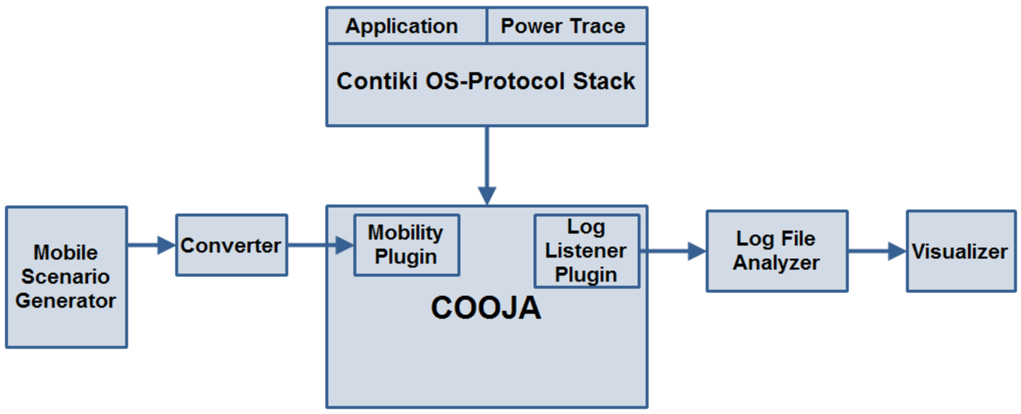


Figure 3.1: A general view of the implementation phase.

In the following sub-sections we give detailed explanation of design and implementation phase.

3.2 Contiki Collect in Mobile Scenarios

Contiki Collect and its TinyOS counterpart show a promising performance in static scenarios. However, they do not show an acceptable performance in mobile scenarios. To adapt Contiki Collect in mobile scenarios, we should pinpoint the reasons for incompatibility of Contiki Collect in mobile networks. Three major reasons behind poor performance of Contiki Collect in mobile networks are as follows.

1. ***Reaction to timeout error:*** When Contiki Collect algorithm sends a packet, it waits for a predefined time (retransmission time) to receive an acknowledgment, otherwise it repeats this process for sometime (set from application layer) before a timeout is invoked. Timeout, in Contiki Collect, increases ETX value to punish the timed out routes but it does not declare the disturbed route as unreachable whereas in mobile scenarios, almost all timeouts happen when the next hop node has already left the communication range of the sender node and the child node is not aware of this change.

2. ***Loop creation:*** Contiki Collect makes use of the same approach in CTP to detect loops. Packets travel through routes that shows reduction in “rtmetric” in each hop. If a node receives a packet from a node that has “rtmetric” less than its own “rtmetric”, it concludes that a loop condition has occurred and sends an acknowledgment packet with setting “rtmetric update” flag to declare that the sender node should update its “rtmetric”. This simple approach is neither agile nor sufficient for mobile scenarios.

3. ***False timeout:*** Intermittent connectivity is a distinguishing feature of mobile sensor networks. This sporadic connection, leads to partitioning of the network graph. Imagine an orphan cluster with some nodes, each of which has some packets in their buffers, enters to the communication range of a node that has a route to a sink node. Having accomplished

neighbor discovery phase, nodes aggressively start to send stored packets in their buffer, similar to the packet streaming. However, nodes compete to access the medium which leads to multiple collisions and timeout. In spite of that, even some nodes fail in the CCA phase where they can not detect a free medium. Saturated links cause a node to issue a timeout to declare a parent-loss while it is in the communication range of its parent.

3.3 Mobile Collect: An Optimized version of Contiki Collect

Previous discussion provided strong grounds for optimization of Contiki Collect. The following sub-sections provides a framework to integrate Contiki Collect in mobile scenarios.

3.3.1 Reaction to Timeout Error

In mobile scenarios, it can be assumed that timeout happens when a target node has disappeared from the communication range of a sender node. Thus, instead of punishing timed out route by increasing its ETX, we simply set "rtmetric" for a timed out route to its maximum value ("RTMETRIC_MAX"). This significant change in the network topology should be advertised to the nodes in the tree branches that the current node is their final parent (new root after timeout). Figure 3.2 shows timeout propagation to the child nodes (since an advertisement is a broadcast packet, all neighbors will hear it). It should be mentioned that timeout advertisements (beacons) are sent by neighbor discovery module as broadcast packets.

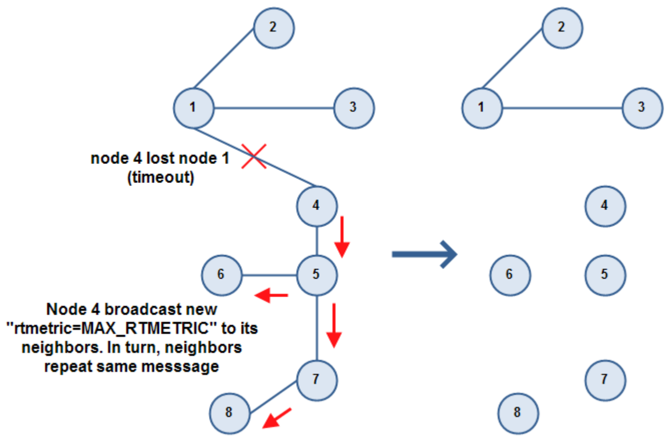

2
1
3
node 4 lost node 1 (timeout)
4
6
5
Node 4 broadcast new "rtmetric=MAX_RTMETRIC" to its neighbors. In turn, neighbors repeat same messsage
7
8
2
1
3
4
6
5
7
8

Figure 3.2: Two snapshots of a network. The left one shows network when timeout occurred. The right one shows network when it reaches steady state.

However, this approach misses other potential parents. For example during tree construction, node 4 was aware about node 3 and it has an entry for node 3 in its routing table. When node 4 loses its parent (node 1), it is possible to switch to node 3 as a new parent. Meanwhile, there is no need to force other nodes (children) to purge their routes. This contribution should be considered carefully because it is highly prone to create loops. Thus, we discuss this approach in the next section.

3.3.2 Detecting and Repairing Loops

Figure 3.3 illustrates typical node operation when timeout occurs. Loop is one of the main design issue in collection protocols especially in mobile scenarios. Loops show more destructive effect on network performance in comparison to static ones. Appropriate parent selection is crucial to prevent loop creation. In other words, the algorithm for parent selection tries to prevent loop creation rather than to repair it.

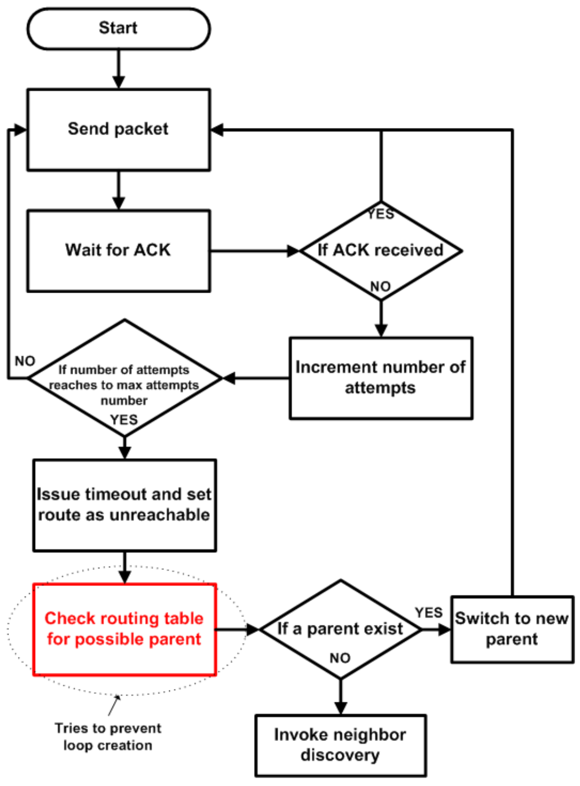

Start
Send packet
Wait for ACK
If ACK received
YES
NO
Increment number of attempts
If number of attempts reaches to max attempts number
NO
YES
Issue timeout and set route as unreachable
Check routing table for possible parent
Tries to prevent loop creation
If a parent exist
YES
NO
Switch to new parent
Invoke neighbor discovery

Figure 3.3: Flowchart of a node operation during sending data packets.

A simple example can clarify a first step in how a loop is created. Imagine in Figure 3.2, node 4 is disconnected from node 1 and node 3 is not in its routing table. Node 4 sets its route to node 1 as unreachable (“rtmetric=RTMETRIC_MAX”) then checks its routing table to find any potential parent. Since “rtmetric” of node 5 is less than “rtmetric” of node 4, it chooses node 5 as a new parent. To prevent creating route in reverse direction, we add current parent of each neighbor to the entries of routing table as well as to the neighbor discovery beacon packets. Accordingly, in the previous example, node 4 does not choose its child as a new parent. This approach is similar to loop handling strategies used by traditional distance vector routing protocols such as RIP[28].

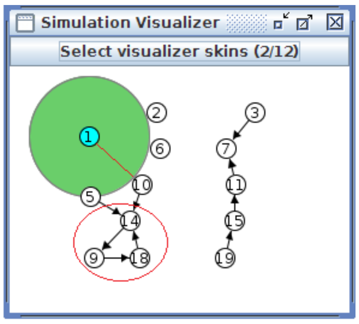


Figure 3.4: An example of loop creation (Triangle loop).

Figure 3.4 shows a triangle loop which happens most of the time during simulation. If each of three nodes are in communication range of another, loop detection will be easy. Also, knowledge about a node's parent is mandatory for the triangle loop detection. Algorithm 3.1 shows the loop avoidance algorithm for the triangle loop.

Algorithm 3.1 Triangle loop detection

```
Require: $selected\_entry \neq null$
1: $addr \Leftarrow address\ of\ current\ node$
2: $loop\_occur \Leftarrow 0$
3: for all $entry \in \{entries\ of\ routing\ table\}$ do
4:     if $entry\_addr \equiv selected\_entry\_parent\_addr$ and $entry\_parent\_addr \equiv addr$ then
5:         $loop\_occure \Leftarrow 1$
6:         return 1
7:     end if
8: end for
9: return 0
```

Neighbor discovery callback[1] in Collect module is another place that is needed to be optimized to mitigate loop creation. Let's first see what kind of problems should be addressed. There are two events that cause a route to be declared as unreachable: timeout due to a physical disconnection and receiving beacon that declares a route should be set as unreachable. To put it in other way, the first event is initialized from a child node whereas the second event is published by the parent. It should be noticed that, the first event ignites the second event. We extend the previous example related to Figure 3.2. We assume that node 4 gets a timeout and there is no parent to switch to (node 3 is not accessible). Hence, node 4 broadcasts a beacon in which it declares that it has lost the route to its parent. Consequently, node 5 receives the beacon and in turn broadcasts its own beacon and declares that it has lost its parent (node 4). Now, if for example node 6 receives the beacon earlier than node 7, it will switch to node 7 as a new parent, and vice versa. Although this example is strikingly similar to the previous one, they come from different origin. Actually, in this example when node 5 advertises its unreachable

[1]When Neighbor Discovery module receives packet from lower layer, it invokes a callback function in Collect module.

state we almost always are confident that both node 6 and 7, eventually, will set their route to node 5 as unreachable. Thus, there is no need to select each other as a parent. We call this algorithm "sibling nodes suppression" which is illustrated in Algorithm 3.2. It should be mentioned that this process will be completed before the parent selection (Figure 3.3) to prevent selecting sibling nodes as parents.

Algorithm 3.2 Sibling nodes suppression

```
1: $parent \Leftarrow \{parent\ of\ current\ node\}$
2: for all $entry \in \{entries\ of\ routing\ table\}$ do
3:     if $parent \equiv \{parent\ of\ entry\}$ then
4:         $entry\_rtmetric \Leftarrow RTMETRIC\_MAX$
5:     end if
6: end for
```

Algorithm 3.3 Parent selection

```
1: $rtmetric \Leftarrow rtmetric\ of\ current\ node$
2: $addr \Leftarrow address\ of\ current\ node$
3: $best \Leftarrow null$
4: for all $entry \in \{entries\ of\ routing\ table\}$ do
5:     $entry\_rtmetric \Leftarrow update\_rtmetric(entry)$
6:     if $entry\_rtmetric < rtmetric$ and
       $entry\_rtmetric \neq RTMETRIC\_MAX$ and
       $entry\_parent\_addr \neq addr$ and
       $entry\_parent\_addr \neq null$ or
       $(entry\_parent\_addr \equiv null$ and $entry\_rtmetric \equiv 0)$ then
7:         if $check\_for\_triangle\_loop(entry) \equiv 0$ then
8:             $best \Leftarrow entry$
9:         end if
10:    end if
11: end for
12: return $best$
```

Although it is desirable to prevent a loop creation rather than to detecting and breaking it, sometimes loop breaking is inevitable. To break a loop, we adhere to the mechanism used in Contiki Collect in which a packet should always experience decreasing ETX in each hop during its travel to a sink node. Otherwise there is a loop in the collection tree. Contiki Collect only sends an acknowledgment packet and request "rtmetric" to be updated (by sending neighbor discovery beacons). However, this approach is not agile enough in mobile scenarios. To avoid persistent loops, in addition to mentioned approach, a problematic route is set to unreachable state ("rtmetric=RTMETRIC_MAX"). If the neighbor discovery finds suitable parent a new route will be established, otherwise the node will be orphaned.

3.3.3 Congestion control: A Simple Solution for Saturated Links

As described earlier, bursty traffic, which is the implication of transient connectivity in mobile scenarios, can lead to timeout while communicating nodes are still in the radio coverage of each other. Mentioned false timeouts stimulate unnecessary neighbor discovery which is power consuming and should be avoided. Utilizing a resource scheduler in which a parent node polls its children for example in a round robin manner and asks each candidate to send a predetermined number of packets could be one possible approach. Decision about the number of packets is challenging and depends on expected QOS (quality of service) policy. However, this algorithm needs some kind of synchronization which should be started by the sink node. In this thesis work such algorithms are left for future work, although it is highly likely they can improve the results. Another solution which is adopted in this thesis is based on traffic shaping (similar to leaky bucket). Specifically, a new field which presents the number of packets in the buffer, has been added to the routing table and to the probe packets (Figure 3.5). Interestingly, every node is aware of the buffer usage of its neighbors as well as that of its potential parent. Having connected to a parent, a child employes a backoff time which is a function of the number of siblings (nodes that are connected to the the same parent) of the current node, their buffer usage as well as that of their parent. Although this simple technique introduces some delay in the packet transmission, it has improved the results. A practical solution could be an algorithm such as StrawMAN [47], but it is out of the scope of this thesis.

3.4 Optimization in MAC Layer

All redesign and optimization which have been done in the previous section should be considered as routing layer optimizations, regardless of possible optimization in MAC layer. In this section possible optimization in MAC layer has been considered.
Originally, Contiki-MAC had been used as the MAC layer protocol during tests and experiments with Mobile Collect. However some ideas such as [32] seems to provide further improvement. In spite of that, Contiki OS contains a collection of MAC protocols with which it is feasible to carry out some experiments. For example: LPP, in contrast to other MAC protocols in Contiki OS, which sends probe packets to check the communication channel could be a possible candidate. Interestingly, it is possible to take advantage of probing for neighbor discovery purposes. However, LPP is not mature enough to be utilized in Contiki Collect in an acceptable way. Moreover, implementation of LPP in Contiki OS suffers from high energy consumption even in static scenarios, let alone mobile scenarios (the power consumption of LPP is around 4-5 times that of Contiki-MAC). In short, there are two limitation for LPP to be well integrated in Contiki Collect: lack of neighbor discovery information in probe and high power consumption. The proposed solutions are discussed in following sub sections.

3.4.1 LPP's Probe: A Neighbor Discovery Tool

LPP is both a generator and a consumer of probe packets. In other words the upper layer is not aware of the contents of a probe packet. Accordingly, a cross layer design is employed to speed up integration of LPP into Contiki Collect. Figure 2.7 illustrates the cross layer interface between Collect and the RDC modules. To send information from LPP to the Collect module, a callback mechanism similar to the one that is described in neighbor discovery has been exploited. During the initial phase of Collect module, a pointer to the Collect's data structure will be sent to LPP. As a result, access to Collect's information will be technically feasible. Figure 3.5 shows a typical LPP probe packet in which neighbor discovery information has been added. The three first fields are originally from default implementation of LPP whereas the four next fields have been added for optimization purpose. The three original fields (default LPP header) are mandatory in the LPP data packet. "Packet type" field reveals that a packet contains data or that is merely a probe packet. "Sender ID" contains the ID of the node that sends the packet. Similarly, the "Receiver ID" field shows which node should accept the packet. It should be mentioned that the "Receiver ID" field for a probe packet is null which shows that a probe is broadcast and can be heard by all nodes. Newly added fields, "rtmetric" and "parent ID" are the same fields as in neighbor discovery message. The "number of packets in buffer" field was discussed previously and "Data Channel" will be explained later.

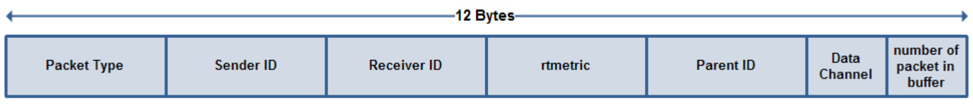


Figure 3.5: A typical LPP probe packet. Each of first five fields is 2 bytes and each of two last fields is 1 byte.

It should be noticed that the mentioned cross layer design does not make the neighbor discovery module useless. In fact, neighbor discovery is mandatory even when a probe contains neighbor discovery information. There are two reasons for necessity of keeping the neighbor discovery module. First, it is used to regulate turning on / off the radio, simply because LPP turns the radio on when there is a pending packet to send. However, even when there is no data packet, probes need to be heard and these two principles contradict each other. Second, although the probe packets are broadcast packets to be received by every node in the communication range, they should be repeated for the duration of a channel check interval. Instead of repeating probe packets, it is possible to send a broadcast packet by the neighbor discovery which forces the radio to be on for the duration of a channel check interval in which each node sends at least one probe and in response receives neighbor discovery's broadcast packets (beacons). Figure 3.6 illustrates the above mentioned description.

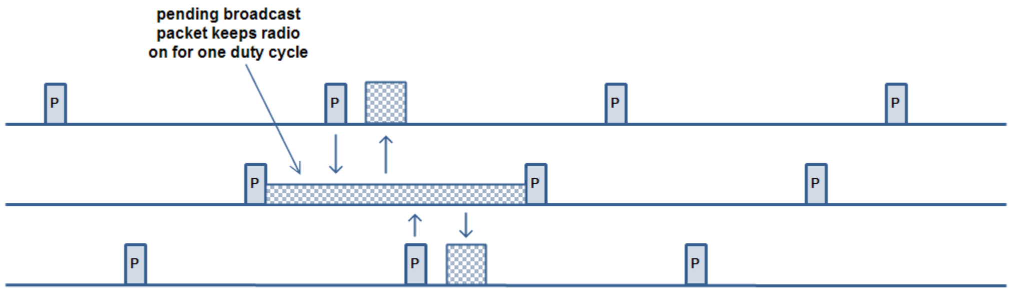


Figure 3.6: A typical packet transmission in LPP. It shows how a broadcast packet can be heard by all nodes in the communication range.

3.4.2 Hardware ACK: A Technique for Reducing Energy Consumption in LPP

This sub-section presents an interesting add-on to the default LPP to alleviate its energy consumption. Prior to proposing any solution for the high power consumption in LPP, the reasons behind that should be revealed. Results from some experiments show that long listening time is the origin of inefficient power consumption. Following formula shows listen and off time during one cycle as well as transmission time for a probe packet.

$$listenTime = \frac{clockSecond}{128}$$

$$offTime = \left(\frac{clockSecond}{channelCheckRate}\right) - listenTime$$

$$Probe\,packet\,transmission\,time = \frac{number\,of\,bits\,in\,probe}{bit\,rate} = \frac{31 * 8}{250000} = 992\,\mu s$$

With “channelCheckRate” equal to 16 Hz, “listenTime” and “offTime” will be 7.8 ms and 54.68 ms respectively. Also, only 992 μs of listen time is used for a probe packet transmission. Decreasing listen time to 3-4 ms, gives power consumption near to that of Contiki-MAC. But, reducing listen time spoils packet reception. Authors of [22] proposed a novel idea by means of hardware ACK of CC2420 to turn off the radio when there is no pending packet. Similarly, the same idea has been exploited in this thesis to improve power consumption performance of LPP. CC2420 has two command strobes[2] for automatic acknowledgment: SACK and SACKPEND. Seemingly, the latter is desirable for our propose because it is possible to declare existence of any pending packet. However, MSPSIM - emulator of TmoteSky - dose not support the

[2]Command strobes may be viewed as a single byte instruction to CC2420 [14].

SACKPEND command. Thus, we adhere to the former one.
When a node sends a probe it waits for a short time interval to receive any possible hardware ACK. Having received the probe, a node with a pending packet, whose receiver address matches the sender address of the probe, sends a hardware ACK. Sending hardware ACK takes one turnaround time (switching from RX to TX or vice versa) which is 12 samples or 192 μs. If the probe sender receives hardware ACK, it keeps its radio in listening mode similar to default LPP, otherwise it quickly turns its radio off. Figure 3.7 visualizes above mentioned functionality.

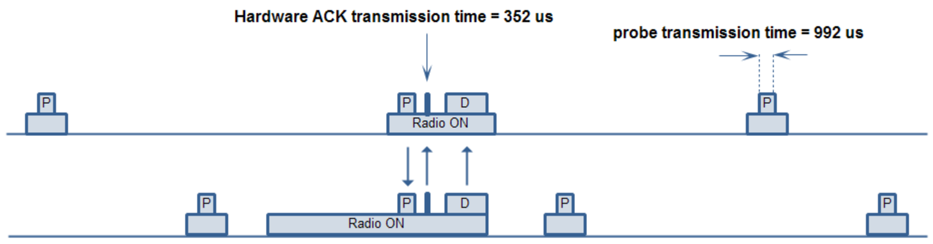

Hardware ACK transmission time = 352 us
probe transmission time = 992 us
P
P
D
Radio ON
P
P
P
D
Radio ON
P
P

Figure 3.7: Integrating hardware ACK into LPP.

Prior to receiving a probe packet, a node should decide whether it is going to send a hardware ACK or not. In other words, since a hardware ACK is issued from the radio chip, decision process can not be delayed until receiving a probe in RDC (LPP) layer. Thus, LPP continuously monitors its sending buffer to see if there is any pending packet or not. In case of a non-empty buffer, it activates the hardware ACK feature of CC2420. Otherwise it will be disabled.
Overreacting [22] is a major drawback for the described approach. Overreacting means a hardware ACK issued from a node with pending data can prolong listen time of other neighbor nodes. This happens as a result of sending a broadcast probe packet requesting a hardware ACK. Authors of [22] have proposed an elegant solution to tackle the overreacting problem. Specifically, when a sender node has a unicast pending data for a receiver node, it enables hardware ACK and address recognition feature of CC2420 as well as sets its hardware address[3] to “receiver address + 0x8000” while the receiver node sends the unicast probe packet to the destination address (receiver address(its own address)+0x8000). As a result, the sender node passes address recognition test whereas other nodes in neighborhood reject the probe packet in the radio chip and quickly turn their radios off.
It is worthwhile to mention that for a broadcast pending packet, address recognition feature should be inactive, as a result of that a broadcaster node is allowed to accept a probe from all nodes and sends a hardware ACK to each of them. Algorithm 3.4 shows the above mentioned mechanism to enable or disable hardware ACK in case of unicast and broadcast packets.

[3]16 bit short address used for address recognition. it is placed in address 0x16B and 0x16A of CC2420 RAM memory [14].

Algorithm 3.4 Enabling or disabling hardware ACK of CC2420

```
1: if $send\_queue\_buffer > 0$ then
2:     $packet \Leftarrow first\_packet\_in\_sent\_queue\_buffer$
3:     if $\{packet\ is\ broadcast\}$ then
4:         $hardware\_ACK\_of\_CC2420 \Leftarrow TRUE$
5:         $address\_recognition\_of\_CC2420 \Leftarrow FALSE$
6:         $short\_address\_of\_CC2420 \Leftarrow 0xFFFF$
7:     else
8:         $hardware\_ACK\_of\_CC2420 \Leftarrow TRUE$
9:         $address\_recognition\_of\_CC2420 \Leftarrow TRUE$
10:        $short\_address\_of\_CC2420 \Leftarrow 0x8000 + node\_ID$
11:    end if
12: else
13:    $hardware\_ACK\_of\_CC2420 \Leftarrow FALSE$
14:    $address\_recognition\_of\_CC2420 \Leftarrow FALSE$
15: end if
```

3.4.3 Multiple Radio Channel: A Technique for Reducing Collision in LPP

Collision is a major problem in WSN. However, collision in receiver-initiated MAC such as LPP is more apparent than sender-initiated MAC, because a receiver-initiated MAC sends probe packets to check the medium while a sender-initiated MAC passively listens to the medium. In scalable and dense networks, where a huge number of nodes send probe packets, collision is escalating. Authors in [22] proposed separating control (probe) channel and data channel to alleviate collision as an optional feature of A-MAC. Similar approach has been adopted to improve the performance of LPP. For LPP, the radio channel 26 has been selected for the probe transmission. During a probe packet creation, LPP inserts a data channel number, which is randomly selected from the interval [11, 25], into the “Data Channel” field of the probe packet (Figure 3.5). As soon as the probe sender receives a hardware ACK, it switches to the selected data channel. On the other side, the “Data Channel” field in the probe packet guides a node with a pending packet to choose an appropriate data channel. It should be noticed that the above mentioned approach is highly dynamic, for example one broadcast packet can be received in different data channels which are assigned by neighbor receiver nodes. Figures 3.8 and 3.9 show a typical data transmission for unicast and broadcast packets. Each line shows the radio activity of each node. Blue and green marks show sending and receiving a packet respectively. The different colors under blue or green marks show on which channel number the radio has been turned on. From Figure 3.8 it can be seen that the first probe has been received, but since address recognition failed, there is no hardware ACK transmission. For the second probe which is received from another node (the target node) address recognition is passed.

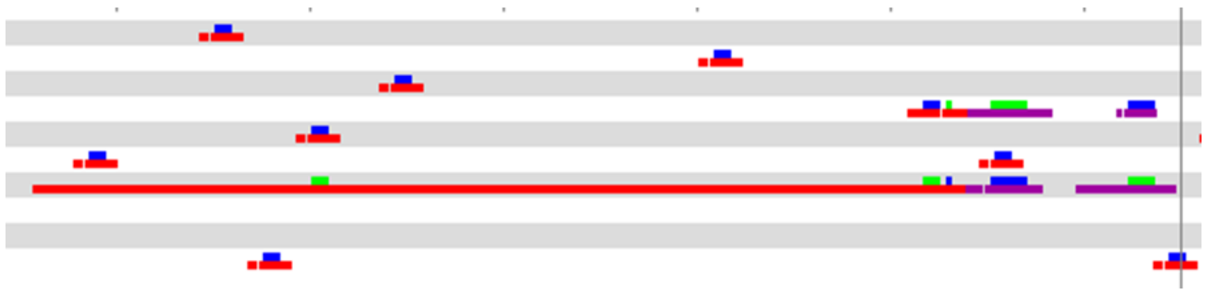


Figure 3.8: Unicast packet transmission in A-MAC. Blue and green colors used to shows sending and reception of packet respectively. Other colors shows channel number.

Figure 3.9 shows a typical broadcast packet transmission. To broadcast a packet, address recognition feature should be disabled to let a broadcast packet be received by all nodes in the neighborhood of the sender.

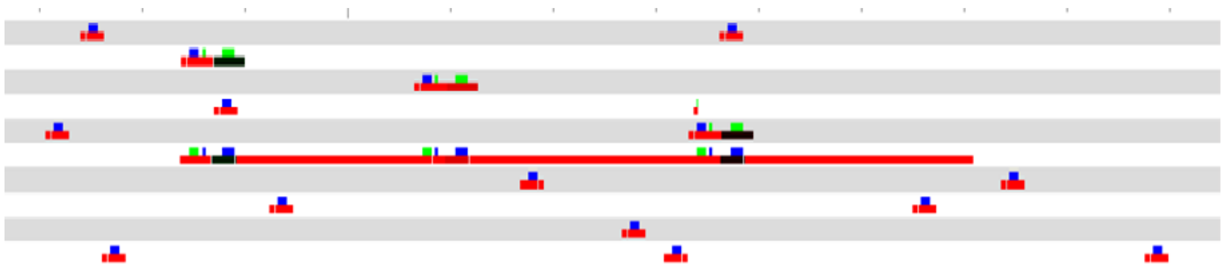


Figure 3.9: Broadcast packet transmission in A-MAC.

Since both LPP and A-MAC share the same background, it is possible to convert LPP to A-MAC. The important point is to find the proper places to add or change the implemented code, after that the rest of the implementation work will be straight forward.

3.5 Developed Tools

This section introduce the tools that are needed for simulation and verification process (Figure 3.1). They can be described as follows.

3.5.1 Mobile Scenario Generator

This block generates a collection of positions for each node. BonnMotion [8], a Java based mobile scenario generator, is the primary tool for this purpose. In spite of its almost comprehensive collection of mobility models, the highway mobility model is missing. However, in this thesis we are interested in highway mobility model which is reasonable enough (among other mobility models) to model rollerblading competition. A partial, yet satisfying highway model has been added to BonnMotion. In implemented highway mobility model we are interested in one dimensional movement. Thus, every node has a fixed position on the y-axis and they move with a random speed on the x-axis. The important options for implemented highway mobility model are as follows.

- **-d:** Declares simulation time (in seconds).
- **-n:** Total number of nodes.
- **-x:** Defines the largest possible value in x-axis (in meter).
- **-y:** Define the largest possible value in y-axis (in meter).
- **-l:** Determines number of lanes.
- **-m:** Minimum speed for one node (in meter per second).
- **-M:** Maximum speed for one node (in meter per second).
- **-s:** Distance (separation) between two lanes (in meter).
- **-r:** Initial distance (separation) between two successive nodes (in meter).

Generated file is not compatible with Mobility plug-in. Thus, we need to convert it to readable format. In this case we should issue following command on generated file [8].

$$bm \;\; IntervalFormat \;\; -l \;\; interval\,length \;\; -f \;\; scenarioFile$$

It is important to choose a reasonable number for "-l" option (default is 1 seconds) to reach a compromise between accuracy and speed.
It should be mentioned that there is another mobility plug-in for COOJA [46] (Contiki OS Java Simulator) called WiseML[4][4] plug-in. Also, we developed a C++ code to convert generated WiseML file to readable format for COOJA. However, since it is not completely supported by some features of COOJA (e.g. in Contiki test editor plug-in), We totally removed it from the implementation phase.

3.5.2 Scenario Analyzer

Scenario analyzer is a C++ code that we have developed to calculate optimal connection time of one node to a sink. It selects one node (source node), then based on transmission range that we have specified selects a possible next hop node and repeats this process to reach to one of the sinks (it consider all possible routes to find one path to one of sinks). The scenario analyzer algorithm repeats this process every 1ms and if it finds that the current route is broken, it tries other possible routes. With this algorithm we can find out about the maximum time duration that

[4]The main goal behind WiseML is to have a description format allowing a standardized storage of traces of experiments.

one node has a connection to a sink during simulation time. For example if one node has zero packet reception, this tool can reveal the reason for that. It should be mentioned that this tool does not consider any routing metric to find optimal routes, it only tries to find a path between a source and a sink.

3.5.3 Log File Analyzer and Visualizing Captured Information

It is possible to use "printf" commands in the Contiki code to lead COOJA to print comments in the log listener plug-in about different situations in code. The generated log file has enormous amount of information. The log file analyzer module, which is a MATLAB code, has been developed to capture this information. Especially, the MATLAB code generates the following information from a log file.

- Number of nodes in a scenario and their node-IDs.
- Number of sink nodes and their node-IDs.
- Number of packets sent by each node.
- Number of packets related to a node which is received by a sink.
- Buffer usage of each node every 10 seconds.
- Power consumption information every 10 seconds. Power consumption is split into listening, transmission, control packets and data packets.

At last, visualizer module, which contains some MATLAB scripts, has been used to generate the required graphs.

Chapter 4

Evaluation

This chapter introduces some conducted experiments with the newly designed algorithm (Mobile Collect). First we give the setting that we have used for these experiments such as mobility models and its parameters, the number of nodes, packet generation rate, etc. Then, we provide some graphs related to reliability, energy consumption and delay. All experiments were carried out by COOJA, a native simulator for Contiki OS, and the results are analyzed by MATLAB.

4.1 Scenario Setups

Collecting data from skaters in rollerblading is the main goal of this thesis. Since we do not have a suitable real trace from rollerblading competition, we have utilized highway mobility model which is the closest synthetic mobility model to rollerblading competition. More than that, we have tested our algorithm with a random waypoint mobility model. Figure 4.1 shows a highway model scenario.

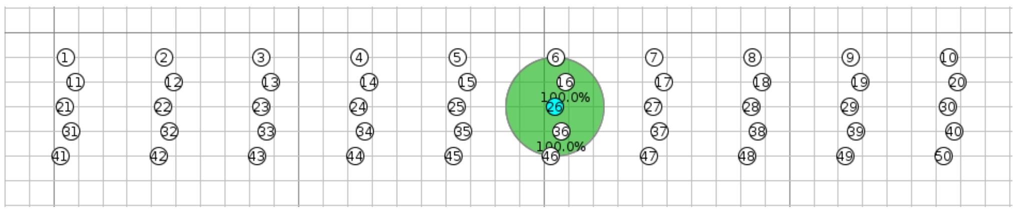


Figure 4.1: A highway mobility model scenario in COOJA.

Based on these two major mobility models, we generated some scenarios based on relative speed of nodes, dense or sparse topology and MAC layer (A-MAC or Contiki-MAC). For all experiments we have considered 50 nodes and the total simulation time is 4000 seconds. Table 4.1 summarizes the setup for experiments. Based on the nodes transmission range and speed there are more sub-experiments.

Experiment	Mobility Model	Nodes Transmission Range (m)	Nodes Speed (m/s)
Mobile Collect vs. Contiki Collect	Highway	5,10,15,20,...,70	[0,0.1] [2,8] [5,15] [10,20]
Mobile Collect vs. Contiki Collect	Random Waypoint	5,10,15,20,...,7	[2,8] [5,15]
Mobile Collect vs. TYMO	Highway	50	[2,8] [5,15]

Table 4.1: Summary of experiments setup.

4.2 Experiment 1: Mobile Collect vs Contiki Collect

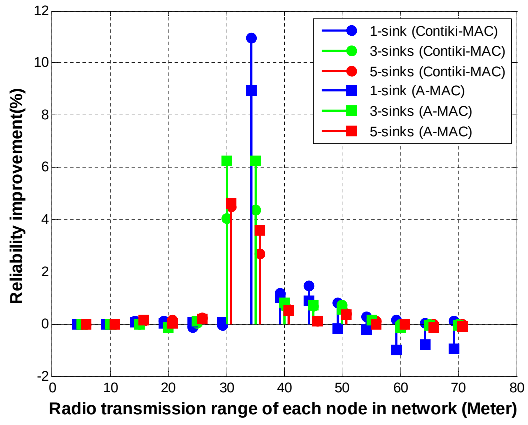
This experiment is divided into two sub-experiments. In the first sub-experiment, Mobile Collect has been tested in a highway mobility model and in the second one its performance has been examined in a random waypoint mobility model. It should be mentioned that in none of the experiments for the Contiki Collect we use A-MAC.

4.2.1 Sub-experiment 1.1: Highway mobility model

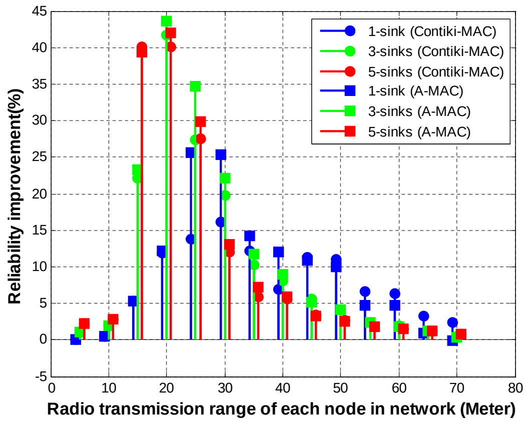
This experiment is closely aligned with the original problem statement of this thesis which is data collection from skiers. We consider four major scenarios each of them has a speed interval, from which a node chooses its own speed randomly. Moreover, every scenario contains 28 sub-scenarios, which contain different densities (can be set by changing radio transmission range) and different MAC protocols (either Contiki-MAC or A-MAC). For Contiki Collect we only consider Contiki-MAC as a MAC protocol, thus there are 14 sub-scenarios for it.

Reliability performance

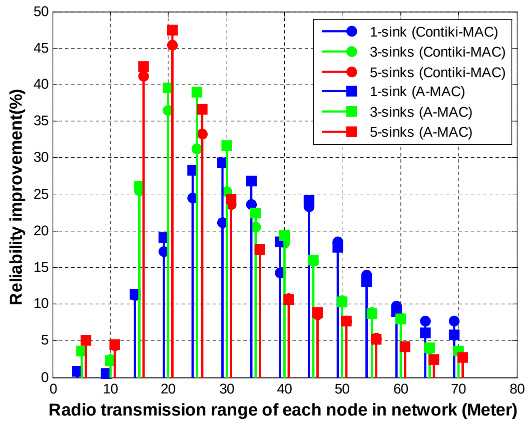
Figure 4.2 shows the difference between reliability of Mobile Collect and Contiki Collect in a highway mobility model. We have considered four sets of mobility with either A-MAC or Contiki-MAC in MAC layer. Figure 4.2a shows improvement in reliability for a very low mobility which is close to a static scenario. In this figure we can see that Mobile Collect with Contiki-MAC shows the same or better performance compared to Contiki Collect. It can be seen from other sub-figures of Figure 4.1 that the scenarios with A-MAC has slightly better reliability performance in the low density area compared to the reliability performance of collection algorithm with Contiki-MAC. Generally, in all sub-figures in high density area, performance of A-MAC has been degraded. Furthermore, we can see that increasing the density leads to decreasing the difference between the reliability performance of the two algorithms. The reason is that an increase in the density, decreases the effect of mobility. In other words, in the area with high density there are more stable links compared to that of the low density area.



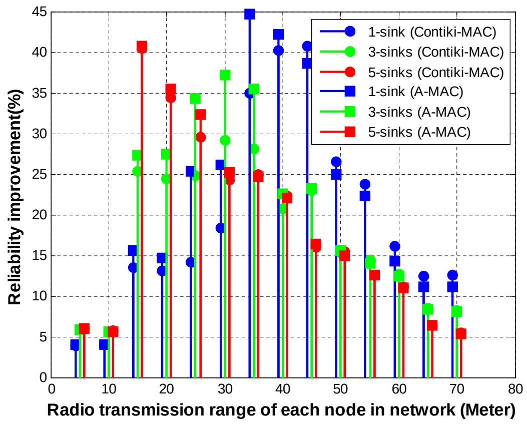
(a) Speed(m/s) = [0,0.1].



(b) Speed(m/s) = [2,8].



(c) Speed(m/s) = [5,15].

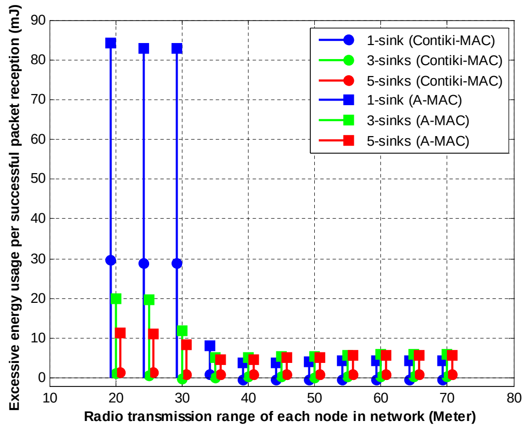


(d) Speed(m/s) = [10,20].

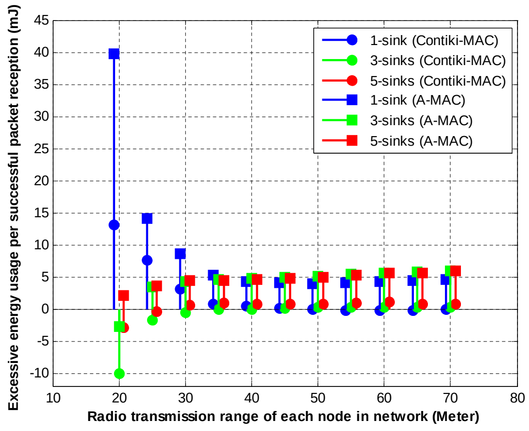
Figure 4.2: Improvement of packet reception in Mobile Collect with highway mobility model.

Energy consumption performance

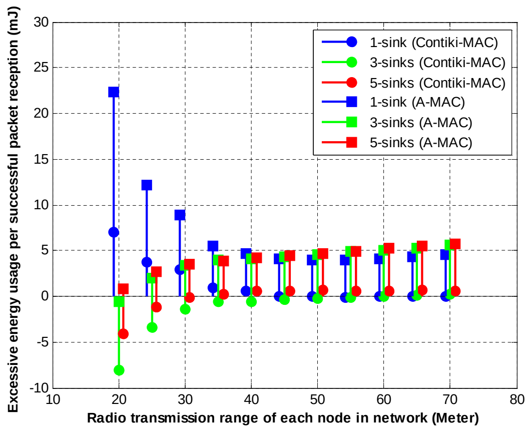
Figure 4.3 shows energy performance which is corresponding to the Figure 4.2. Both collect algorithms (Mobile Collect and Contiki Collect) show the same energy performance in all scenarios with different mobility, density and sink nods number. This is an interesting result that Mobile Collect can reach a higher reliability while it keeps energy consumption per packet in the previous level. In spite of that, Mobile Collect with A-MAC consume approximately 5 mJ extra energy per packet to achieve almost the same reliability performance as Contiki-MAC.



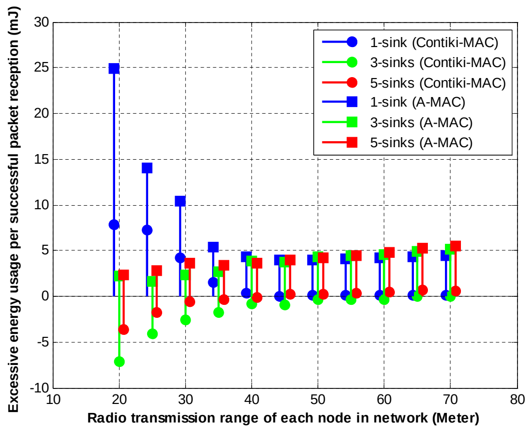
(a) Speed(m/s) = [0,0.1].



(b) Speed(m/s) = [2,8].



(c) Speed(m/s) = [5,15].



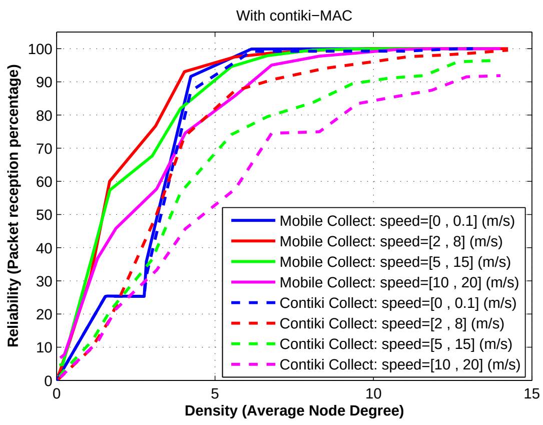
(d) Speed(m/s) = [10,20].

Figure 4.3: Excessive energy usage in Mobile Collect with highway mobility model.

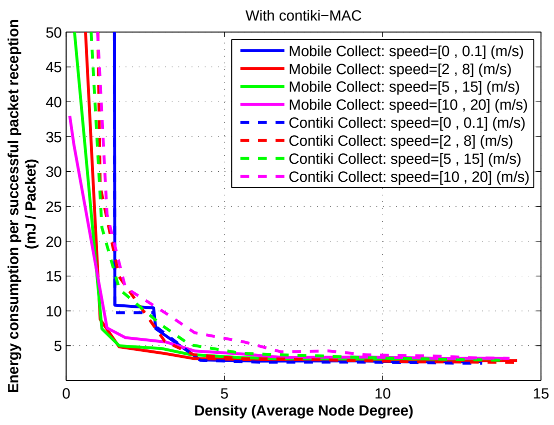
It is clear from Figure 4.3 that for transmission range of 5, 10 and 15 meters we have high energy consumption per packet. We could ignore these result, because packet reception in these area is near to zero and per packet energy will be high.

An example of detailed statistics

Figure 4.4 shows a detailed information (not difference or improvement) for the scenarios with 3 sinks and Contiki-MAC as a MAC protocol. We have used average node degree in x-axis which is calculated by BonnMotion based on the information from different transmission ranges.



(a) Reliability performance.

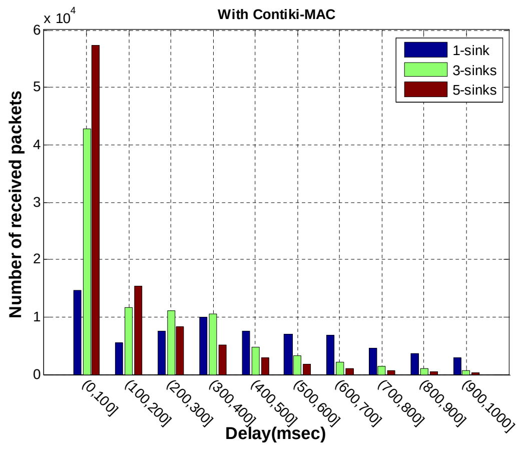


(b) Energy consumption performance.

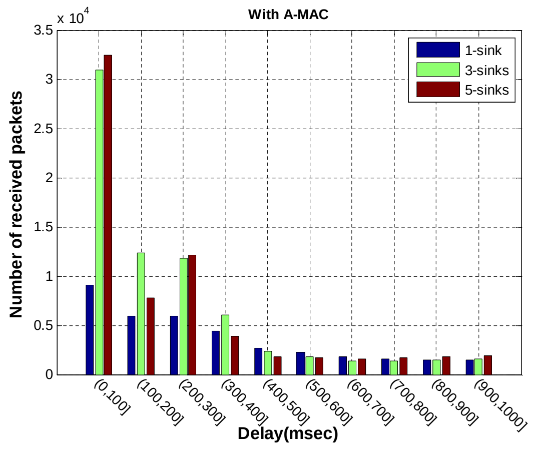
Figure 4.4: Reliability and energy performance in Mobile and Contiki Collect with highway mobility model (3-sinks and Contiki-MAC).

Delay performance: Sink numbers effect

Figures 4.5a and 4.5b show delay performance histograms for Mobile Collect with Contiki-MAC and A-MAC respectively. We calculate the delay based on the difference between receiving and sending time (some packets are not received at all, so delay is meaningless for them). For these two figures we studied the effect of sink nodes number on performance. From these two graphs we can see that majority of packets have delay between 0 to 100 ms. Moreover, increase in the number of sink nodes improve delay performance. This could be due to less hops and more steady links. Additionally, it is clear from these two graphs that Contiki-MAC has better delay performance compared to A-MAC.



(a) Delay performance with Contiki-MAC.

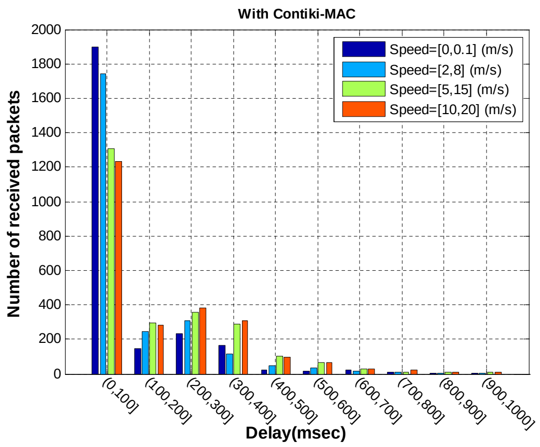


(b) Delay performance with A-MAC.

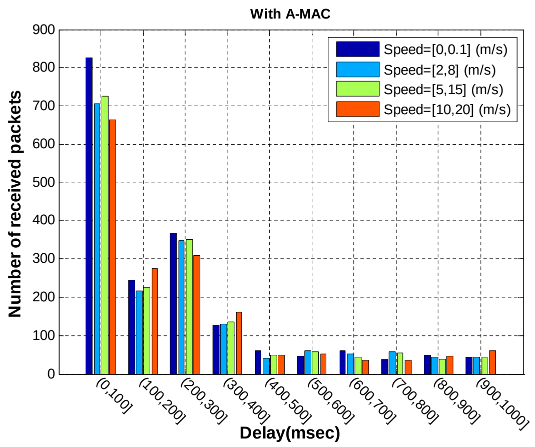
Figure 4.5: Delay performance of Mobile Collect with different number of sinks.

Delay performance: Velocity effect

Figure 4.6a and 4.6b give information about delay performance in Mobile Collect with Contiki-MAC and A-MAC respectively. In contrast to the previous figures, we consider the effect of node speed on the delay performance. It can be seen that low speed scenarios have better delay performance. This result is quite sensible, because in the low speed scenarios, collection tree needs less update in comparison with the high speed scenarios. Similar to the previous graph, we can see that Contiki-MAC shows better delay performance compared to A-MAC.



(a) Delay performance with Contiki-MAC.



(b) Delay performance with A-MAC.

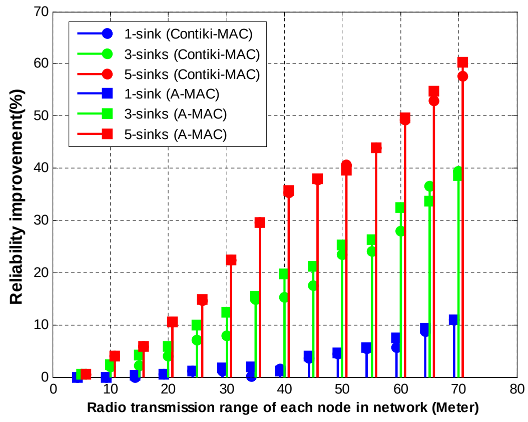
Figure 4.6: Delay performance of Mobile Collect with 3 sinks and different speed intervals.

4.2.2 Sub-experiment 1.2: Random waypoint mobility model

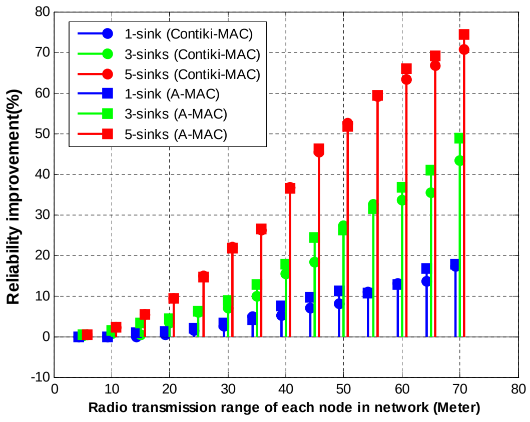
In this sub-section we consider performance of Mobile Collect in random waypoint mobility model. For this mobility model we consider pause time equal to zero. We have evaluated Mobile Collect with two different set of velocity for both A-MAC and Contiki-MAC.

Reliability performance

Figures 4.7a and 4.7b demonstrate improvement in the reliability performance for the node speed of [2,8] and [5,15] respectively. In these figures we consider the difference between reliability of Mobile Collect and Contiki Collect to show how much the new algorithm has improved. Interestingly Contiki Collect shows very small packet reception (Figure 4.9a) while Mobile Collect shows a better performance. The main reason could be the reaction to timeout error — the first problem of Contiki Collect in mobile scenarios introduced in section 3.2. However, Mobile Collect has an agile response to the transient links which dominate in random mobility models. An increase in the number of sinks has a major contribution to the reliability performance. As mentioned before, increase in the radio transmission range which is corresponding to the increase in network density alleviates the effect of random mobility of nodes, thus a high density means less transient links and a better packet reception.



(a) Improvement in reliability performance for speed(m/s)=[2,8].

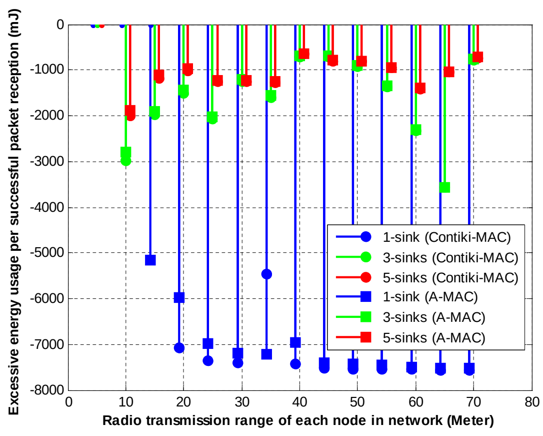


(b) Improvement in reliability performance for speed(m/s)=[5,15].

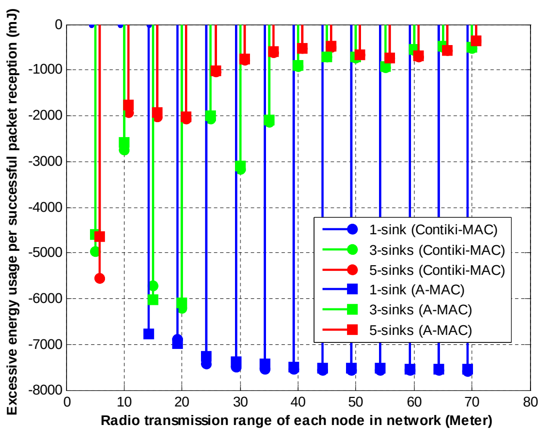
Figure 4.7: Improvement of packet reception in Mobile Collect with random waypoint mobility model.

Energy consumption performance

Figures 4.8a and 4.8b show energy performance for two different sets of velocity. Similar to the previous experiment we are interested to see how much extra per packet energy has been consumed in the new algorithm compared to the Contiki Collect. All per packet energy for different densities values are negative which shows that Contiki Collect consumed energy only to send beacons and that the outcome of the beacon packets is almost nothing. Interestingly, in these two figures it is clear that the two MAC protocols, A-MAC and Contiki-MAC, have almost the same energy performance. This also can be seen from Figure 4.9b.

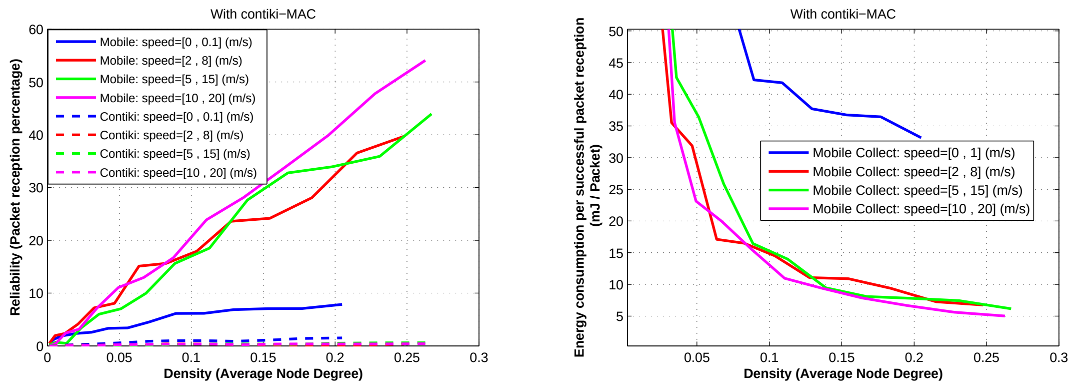


(a) Energy consumption performance for speed(m/s)=[2,8].



(b) Energy consumption performance for speed(m/s)=[5,15].

Figure 4.8: Excessive energy usage in Mobile Collect with random waypoint mobility model.



(a) Reliability performance. **(b)** Energy consumption performance.

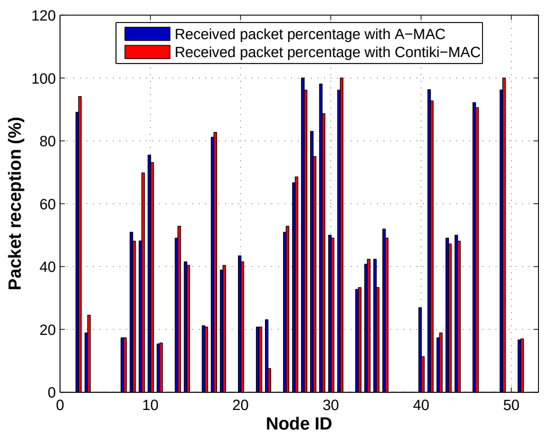
Figure 4.9: Reliability and energy performance in Mobile and Contiki Collect with the random mobility model (3-sinks and Contiki-MAC).

4.2.3 An example of micro-benchmark

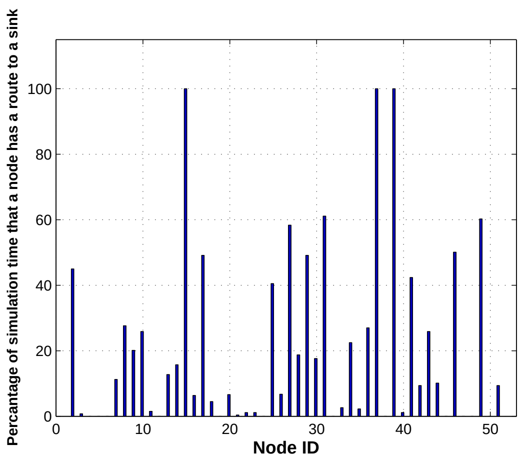
In the previous figures we considered macro-benchmark graphs which deal with the whole network, In these graphs it is not possible to get detailed information about each node in the network. In this section some figures which provide micro-benchmark have been illustrated. Usually, these graphs are suitable during algorithm development and testing. In some cases they contributed to finding bugs in the algorithm.

Reliability and ideal connection time

Figure 4.10 shows reliability performance for each node in the network. From Figure 4.10a we can see some nodes have small or zero percent packet reception (by a sink node). To find the reason we can use the tool that has been introduced in sub-section 3.5.2. Figure 4.10b shows ideal connection time, the output of mentioned tool, of each node. With these two figures we can detect unexpected behavior in the algorithm. If there is a high connection time, high packet reception rate for those nodes is also expected. For example, the second node from left has very low ideal connection time, but sink nodes received almost 20% of sent packets. This is considered as an excellent reliability performance. In spite of that, due to multihop and mobile nature of the scenario we cannot have a strict judgment about reliability based on ideal connection time figures. However, it could be one way to verify designed algorithm. It should be mentioned that the nodes with 100% connection time and zero packet reception are sink nodes.



(a) Reliability performance.

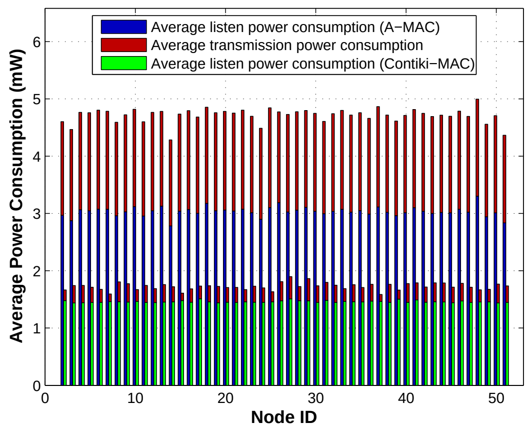


(b) Ideal connection time.

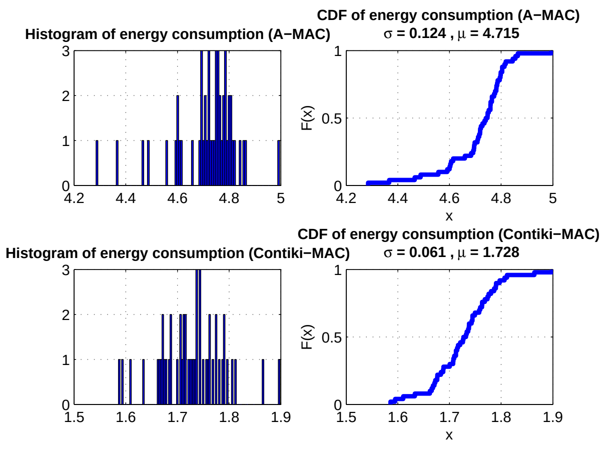
Figure 4.10: Reliability and ideal connection time. 50 nodes, 3 of them sinks with random speed from [5,15] interval in the highway mobility model.

Energy consumption performance

Figure 4.11a shows energy consumption per node. Also, this figure provides more detailed information about the origins of energy consumption which is the sum of listen (receive) and transmission energy consumption. Listen energy consumption could be a packet reception or idle listening. Furthermore, energy consumption due to the probing in A-MAC is considered as the listen energy consumption.



(a) Energy consumption performance.



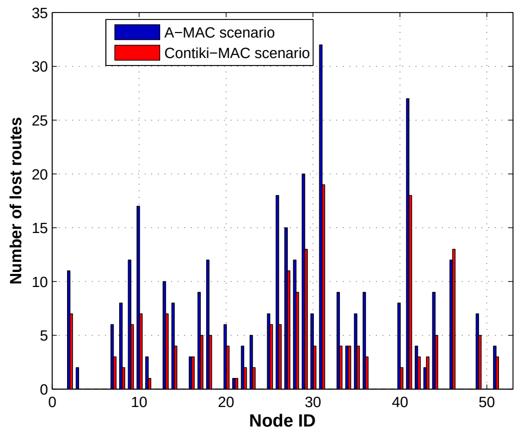
(b) Power consumption histogram and corresponding CDF.

Figure 4.11: Energy consumption performance. 50 nodes, 3 of them sinks with random speed from [5,15] interval in the highway mobility model.

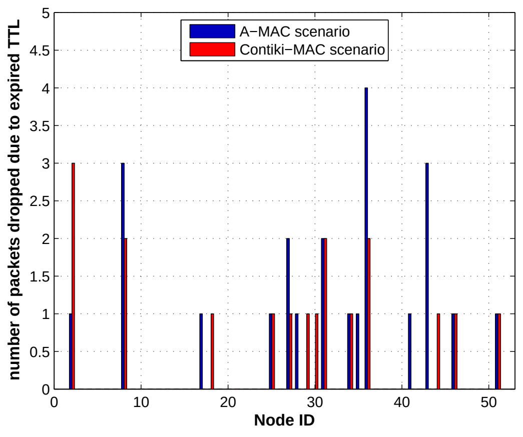
Figure 4.11b illustrates histogram and CDF (Cumulative Distribution Function) related to Figure 4.11a. It is clear from both figures that collection algorithm based on A-MAC has high energy consumption compared to its Contiki-MAC counterpart. Moreover, the standard deviation of energy consumption in A-MAC is slightly higher than Contiki-MAC. In other words, some nodes, have more energy consumption compared to the others.

Number of lost routes and dropped packet due to the expired TTL

Figure 4.12a shows the number of lost routes for each node in the network. “Lost-routes” is a counter that increment when a node fails to have access to its parent. It is highly depended on the mobility and the network dynamics.



(a) Number of lost routes.



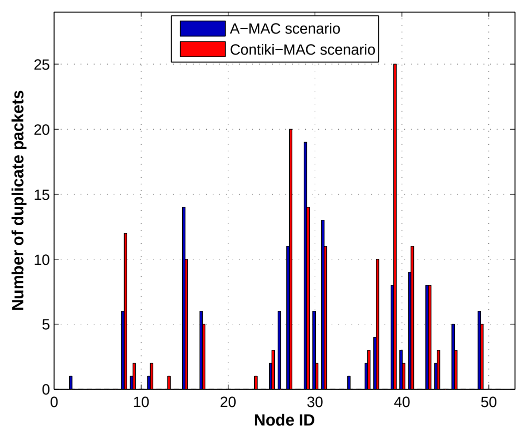
(b) Number of dropped packet due to expired TTL.

Figure 4.12: Number of lost routes and dropped packet due to expired TTL. 50 nodes, 3 of them sinks with random speed from [5,15] interval in the highway mobility model.

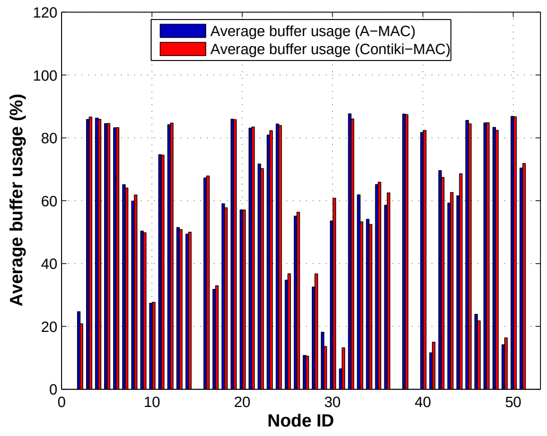
Figure 4.12b gives information about the number of dropped packets due to the expired TTL (Time to Live) which is set to 15. As can be seen from this figure, there are a few dropped packets due to the expired TTL. This graph could be one way to examine the resistance of an algorithm to the loop creation which is the core challenge for data collection algorithm especially in mobile scenarios.

Number of duplicate packets and buffer usage performance

Figure 4.13a shows the number of duplicate packets received by each node in a typical mobile scenario. Duplicate packets are the consequence of the lost ACK and mobility is the main reason for the lost ACK.



(a) Number of duplicate packets.



(b) Buffer usage performance.

Figure 4.13: Number of duplicate packets and buffer usage performance. 50 nodes, 3 of them sinks with random speed from [5,15] interval in highway mobility model.

Figure 4.13b shows average buffer usage for each node. It can be seen that, generally nodes have high buffer usage which shows that most of the time they have to store packets in their buffer. High buffer usage matches Figure 4.10b.

Detailed analysis of node 14

Figure 4.14 demonstrates a detailed graph for node 14 as an example. It shows four statistics of node 14: buffer usage, dropped packets due to expired TTL, the number of lost routes and duplicate packets during simulation time. Again, we can use this graph to analyze an individual node's behavior. For example in region 1 of Figure 4.14 (A-MAC) there is a quick drop in buffer usage which means a new parent has been discovered, but in region 2 there is an increase in the buffer size and also one step increase in the number of lost routes in which the node has lost its parent. In region 3 (the number of lost routes) we can see several connecting nodes and disconnecting from the parent that leads to some duplicate packets which are the result of lost ACK. One explanation could be saturated links which leads to consecutive connection and disconnection. Another explanation could be that two nodes (parent and child) are in critical point (the circles of transmission range of both nodes are tangent to each other) for some time.

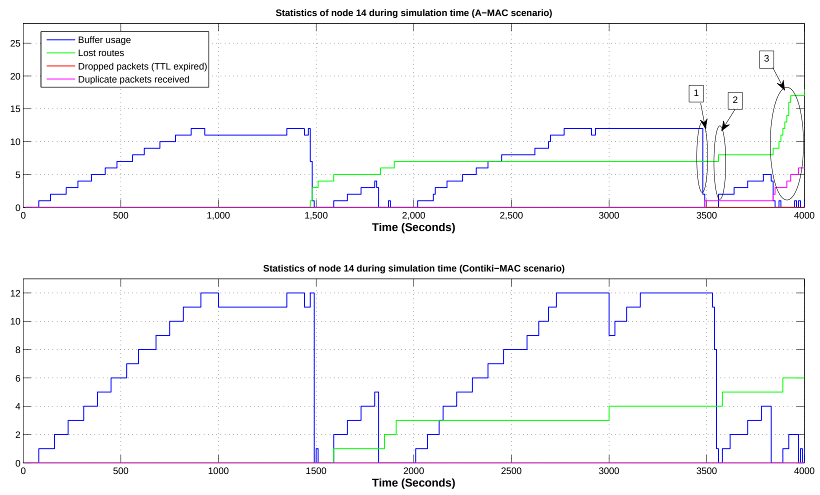


Figure 4.14: Detailed analysis of node 14.

4.3 Experiment 2: Mobile Collect vs DYMO

DYMO is a MANET routing algorithm. It is very similar to AODV routing algorithm. We have tested TYMO, an implementation of DYMO in TinyOS, with duty cycled radio, but the result, even in static scenarios, was not satisfying. Thus, for the experiment that we have used TYMO, we do not consider duty cycled radio as well as energy consumption performance for TYMO. In this experiment we consider 50 nodes, one of them is a sink node. All nodes are moving according to the highway mobility model. Table 4.2 gives the result for this experiment.

Algorithm	Reliability (%)
Mobile Collect	98.9
TYMO	55.8

(a) Speed (m/s) = [5,15]

Algorithm	Reliability (%)
Mobile Collect	99.2
TYMO	66.2

(b) Speed (m/s) = [2,8]

Table 4.2: Mobile Collect vs. DYMO.

It can be seen from Table 4.1 that high speed degrades reliability performance of DYMO while Mobile Collect can preserve its reliability in both speed intervals.

Chapter 5

Conclusions and Future work

5.1 Conclusions

Data collection from a large number of sensor nodes is a widespread application for wireless sensor networks. There are extensive research in data collection algorithms which are suitable for stationary sensor nodes or at most they consider small dynamics in the network topology. However, there is an increasing demand for mobility-aware algorithms which stem from applications that deals with inherently mobile objects. Applications in sports such as Marathon, rollerblading and cycling are examples of scenarios in which sensor nodes are attached to athletes that are intrinsically mobile.

In this thesis we studied data collection in a mobile wireless sensor network where all nodes, the sources and the sinks, are mobile. We started with Contiki Collect to understand its performance in mobile scenarios. We optimized Contiki Collect by adding an agile and reactive solution to physical link disconnection which is the result of nodes mobility. Also, we optimized parent selection algorithm to have loop free data routing. The result shows considerable improvement in the reliability performance for sparse topologies while it keeps the same performance for static scenarios. This is achieved with only slight increase in energy consumption.

In the next phase of thesis we implemented A-MAC, a state-of-the-art receiver-initiated MAC protocol, to test its performance with the new algorithm (Mobile Collect). The result shows a slight improvement or the same performance as Contiki-MAC while there is a considerable increase in energy consumption in A-MAC case.

Furthermore, Mobile Collect shows a significant improvement in the reliability performance compared to the MANET routing algorithm (TYMO). In spite of that, Mobile Collect takes advantage of duty cycled radio to save energy, but TYMO is designed for non-duty cycled radio.

5.2 Future work

- Ideally simulation work should be done based on the real traces. In this thesis we primarily are interested in data collection from skaters in rollerblading competition. Due to the lack of real traces we emulated skiers motion according to the highway mobility model.

- A small percentage of sensor nodes should be sink nodes. However, we need a dimensioning process to detect appropriate number of sinks. In this case, also, we need to have access to real traces of the scenario for proper dimensioning.

- It is needed to use a congestion control algorithm alongside Mobile Collect. It could alleviate bursty traffic generated by the nodes with full buffer during connection to the parent node.

- We have utilized FIFO (first in first out) discipline for the buffers of the nodes. We could have more mature buffer management strategy. For example packets that are in the middle of their path should be sent quickly.

- Using RSSI (Received Signal Strength Indicator) in the routing metric could be another improvement.

- Having added GPRS connection quality to the routing metric, less nodes would be connected to the sink with poor GPRS connection quality.

- The designed algorithm should be evaluated on a mobile testbed.

Bibliography

[1] Micaz datasheet. www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf.

[2] Sun small programmable object technology (sun spot) : Theory of operation. http://sunspotworld.com/docs/Purple/SunSPOT-TheoryOfOperation.pdf.

[3] Tmote sky datasheet. http://sentilla.com/files/pdf/eol/tmote-sky-datasheet.pdf.

[4] Wisebed project: Deliverable d4.1 first set of well-designed simulations experiments and possible benchmarks version 2009. http://www.wisebed.eu/images/stories/deliverables/d4.1.pdf/.

[5] IEEE standard for information technology- telecommunications and information exchange between systems- local and metropolitan area networks- specific requirements part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (wpans). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, 2006.

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

[7] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102–114, 2002.

[8] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn. Bonnmotion: a mobility scenario generation and analysis tool. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, pages 1–10. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010.

[9] F. Bai and A. Helmy. A survey of mobility models. *Wireless Adhoc Networks. University of Southern California, USA*, 206, 2004.

[10] M. M. Bin Tariq, M. Ammar, and E. Zegura. Message ferry route design for sparse ad hoc networks with mobile nodes. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 37–48. ACM, 2006.

[11] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 307–320. ACM, 2006.

[12] N. Burri, P. Von Rickenbach, and R. Wattenhofer. Dozer: ultra-low power data gathering in sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, pages 450–459. ACM, 2007.

[13] A. Chakrabarti, A. Sabharwal, and B. Aazhang. Using predictable observer mobility for power efficient design of sensor networks. In *Proceedings of the 2nd international conference on Information processing in sensor networks*, pages 129–145. Springer-Verlag, 2003.

[14] Chipcon AS. CC2420 Datasheet (rev. 1.3), 2005.

[15] Douglas S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. *Wireless Networks*, 11(4):419–434, 2005.

[16] Crawdad. Community resource for archiving wireless data at dartmouth, 2006. `http://crawdad.cs.dartmouth.edu/`.

[17] M. Demirbas, O. Soysal, and A. Ş. Tosun. Data salmon: A greedy mobile basestation protocol for efficient data collection in wireless sensor networks. In *Proceedings of the 3rd IEEE international conference on Distributed computing in sensor systems*, pages 267–280. Springer-Verlag, 2007.

[18] I. Demirkol, C. Ersoy, and F. Alagoz. Mac protocols for wireless sensor networks: a survey. *Communications Magazine, IEEE*, 44(4):115–121, 2006.

[19] Mario Di Francesco, Sajal K. Das, and Giuseppe Anastasi. Data collection in wireless sensor networks with mobile elements: A survey. *ACM Trans. Sen. Netw.*, 8:7:1–7:31, August 2011.

[20] A. Dunkels, B. Grönvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the First IEEE Workshop on Embedded Networked Sensors*, Tampa, Florida, USA, November 2004.

[21] A. Dunkels, L. Mottola, N. Tsiftes, F. Osterlind, J. Eriksson, and N. Finne. The announcement layer: Beacon coordination for the sensornet stack. *Wireless Sensor Networks*, pages 211–226, 2011.

[22] P. Dutta, S. Dawson-Haggerty, Y. Chen, C. J. M. Liang, and A. Terzis. Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 1–14. ACM, 2010.

[23] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G. S. Ahn, and A. T. Campbell. Bikenet: A mobile sensing system for cyclist experience mapping. *ACM Transactions on Sensor Networks (TOSN)*, 6(1):6, 2009.

[24] A. El-Hoiydi, J.D. Decotignie, C. Enz, and E. Le Roux. Poster abstract: wisemac, an ultra low power mac protocol for the wisenet wireless sensor network. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 302–303. ACM, 2003.

[25] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 29–39. ACM, 2008.

[26] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 1–14. ACM, 2009.

[27] J. Hallberg, S. Svensson, A. Östmark, P. Lindgren, K. Synnes, and J. Delsing. Enriched media-experience of sport events. In *Sixth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04)*, Lake District National Park, United Kingdom, December 2004.

[28] C. L. Hedrick. RFC 1058: Routing information protocol, June 1988.

[29] D. Jea, A. Somasundara, and M. Srivastava. Multiple controlled mobile elements (data mules) for data collection in sensor networks. *Distributed Computing in Sensor Systems*, pages 244–257, 2005.

[30] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In *Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*, pages 96–107. ACM, 2002.

[31] M. Keally, G. Zhou, and G. Xing. Sidewinder: a predictive data forwarding protocol for mobile wireless sensor networks. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON'09. 6th Annual IEEE Communications Society Conference on*, pages 1–9. IEEE, 2009.

[32] Olaf Landsiedel and Mikael Johansson. Poster abstract: Towards a life without link estimation. In *SenSys'10: Proc. of the 7th ACM Conference on Embedded Networked Sensor Systems*, 2010.

[33] J. W. Lee, B. Kusy, T. Azim, B. Shihada, and P. Levis. Whirlpool routing for mobility. In *Proceedings of the eleventh ACM international symposium on Mobile ad hoc networking and computing*, pages 131–140. ACM, 2010.

[34] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation-Volume 1*, pages 2–2. USENIX Association, 2004.

[35] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3):19–20, 2003.

[36] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali. Routing without routes: The backpressure collection protocol. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 279–290. ACM, 2010.

[37] D. Moss and P. Levis. Box-macs: Exploiting physical and link layer boundaries in low-power networking. Technical report, Technical Report SING-08-00, Stanford University, 2008.

[38] R. Musaloiu-E, C. J. M. Liang, and A. Terzis. Koala: Ultra-low power data retrieval in wireless sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 421–432. IEEE Computer Society, 2008.

[39] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 95–107. ACM, 2004.

[40] I. Rhee, A. Warrier, M. Aia, J. Min, and M.L. Sichitiu. Z-mac: a hybrid mac for wireless sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 16(3):511–524, 2008.

[41] Kyle Jamieson Rodrigo Fonseca, Omprakash Gnawali and Philip Levis. Tep 119: Collection. www.tinyos.net/tinyos-2.x/doc/pdf/tep119.pdf.

[42] Kyle Jamieson Sukun Kim Philip Levis Rodrigo Fonseca, Omprakash Gnawali and Alec Woo. Tep 123: The collection tree protocol (ctp). www.tinyos.net/tinyos-2.x/doc/pdf/tep123.pdf.

[43] R.C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2-3):215–233, 2003.

[44] A. A. Somasundara, A. Kansal, D. D. Jea, D. Estrin, and M. B. Srivastava. Controllably mobile infrastructure for low energy embedded networks. *IEEE Transactions on Mobile Computing*, pages 958–973, 2006.

[45] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259. ACM, 2005.

[46] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *Proceedings of the First IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2006)*, Tampa, Florida, USA, November 2006.

[47] F. Österlind, N. Wirström, N. Tsiftes, N. Finne, T. Voigt, and A. Dunkels. StrawMAN: Making Sudden Traffic Surges Graceful in Low-Power Wireless Networks. In *Proceedings of the 2010 ACM HotEMNETS Workshop on Hot Topics in Embedded Networked Sensosr*, Killarney, Ireland, June 2010.

[48] Y. Sun, O. Gurewitz, and D. B. Johnson. RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 1–14. ACM, 2008.

[49] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, New York, NY, USA, June 2002.