Protocol Design of Sensor Networks for Wireless Automation

PAN GUN PARK

KTH Signals
Sensors and Systems

Master's Degree Project
Stockholm, Sweden 2007

Protocol Design of Sensor Networks for Wireless Automation

PAN GUN PARK

KTH Signals
Sensors and Systems

Master's Degree Project

January 2007

Automatic Control Group
School of Electrical Engineering

Abstract

The recent development of control applications over Wireless Sensor Networks (WSNs) imposes new approaches to the protocol design. These networks are characterized by the scarcity of energy supply and processing capabilities. Furthermore, existing protocol solutions are often based on the traditional OSI model, where communication layers are not optimized to support efficiently the reliability and latency requirements imposed by control applications. The critical aspects of wireless transmission have lead to a lack of protocols that are able to guarantee latency and quality of service under unreliable channel conditions.

In this thesis, we design and implement a cross-layer protocol for WSNs in industrial automation, the Extended Randomized Protocol, which considers jointly physical layer aspects (as power control and duty cycling strategies), randomized MAC and routing. The protocol can be considered and extension of an already existing Randomized Protocol, and it is designed with the objective to maximize the network lifetime under the constraints of error rate and end-to-end delay in the packet delivery.

As a relevant part of our activity, we have provided a complete test bed implementation of the protocol building a WSN with TinyOS and a large number of Moteiv's Tmote Sky wireless sensors. An experimental campaign has been conducted in order to test the validity of the protocol solution we propose. Experimental results show that the protocol achieves the required successful packet reception rate and the latency constraints while minimizing the energy consumption. Despite the fact that improving solutions are necessary to take into account the problem of duplicated packets, our protocol solution seems to be a good candidate for WSN in industrial automation.

Introduction

The breakthrough of micro-electro-mechanical system technology, wireless communication, digital electronics make it possible to develop sensor nodes that are small, inexpensive, low-power and has communication function in short distance. Wireless Sensor Networks (WSNs) are wireless networks comprising of spatially distributed sensor nodes to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different applications. In order words, WSNs are an attractive means to monitor environment and to come true the ubiquitous society.

Although WSNs promises the industrial automation, there is no standard protocol providing latency and quality of service. The main issue is the reliability of the communication from randomness and time-varying characteristics of wireless channel. Furthermore, WSNs necessitate energy-efficient communication protocols because of severe energy constraints. However, the most of the existing solutions are based on classical layered protocols approach, without a clear explanation of how these melt into a optimal solution. Considering the scarce energy, processing resources, reliability of WSN, joint optimization and design of networking layers, i.e., unified cross-layer design, is the one of most promising alternative to traditional layered protocol architectures, i.e., OSI model.

The thesis can be divided into four parts: the first part is introduction to WSNs (Chapter 1). The second part contains the explanation of Extended Randomized Protocol (Chapter 2) and comment about the extended points to the Randomized Protocol (Chapter 3). The third part includes a description of how to set up the experiment (Chapter 4) and discussion of experimental results (Chapter 5), and finally the conclusion, achievement, and future work (Chapter 6).

Contents

List of Tables

List of Figures

Chapter 1

Overview of Wireless Sensor Networks

1.1 Application and Challenges on WSNs

Sensors are devices that produce measurable responses to a change in a physical condition like temperature or pressure. The wireless communication channel provides a medium to transfer signals from sensors to exterior world or a computer network, and also a mechanism of communication to establish and maintenance of Wireless Sensor Networks (WSNs), leverage the idea of sensor networks based on collaborative effort of a large number of nodes. Each sensor node is capable of only a limited amount of processing. But when coordinated with the information from a large number of other nodes, they have the ability to measure a given physical environment in great detail. Thus, WSNs are an increasingly attractive means to bridge the gap between the physical and the virtual world [1] [2].

The WSNs represents a new monitoring and control capability for applications such as industries, transportation, manufacturing, health care, environmental oversight, safety and security [3]. For example, the physiological data about a patient can be monitored remotely by a doctor. While this is more convenient for the patient, it also allows the doctor to better understand the patient's current condition. Sensor networks can also be used to detect foreign chemical agents in the air and the water. They can help to identify the type, concentration, and location of pollutants. In essence, sensor networks will provide the end user with intelligence and a better understanding of the environment. We believe that, in future, wireless sensor networks will be an integral part of our lives, more so than the present-day personal computers.

These wide range of applications for WSNs can be classified into three cat-

egories [3]:

- monitoring space;
- monitoring things;
- monitoring the interactions of things with each other and the encompassing space;

The first classification includes environmental and habitat monitoring, precision agriculture, indoor climate control, surveillance, treaty verification, and intelligent alarms. The second includes structural monitoring, ecophysiology, condition-based equipment maintenance, medical diagnostics, and urban terrain mapping. The most dramatic applications involve monitoring complex interactions, including wildlife habitats, disaster management, emergency response, ubiquitous computing environments, asset tracking, health care, and manufacturing process flow.

In spite of the diverse applications, WSNs pose a number of unique technical challenges different from traditional wireless ad hoc networks [2] [4] [5]. Therefore protocols and algorithms that have been proposed for traditional wireless ad hoc networks, are not well suited for the unique features and application requirements of sensor networks. To illustrate this point, the challenges or the differences between sensor networks and traditional networks are outlined below:

- The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network:

  Since large number of sensor nodes are densely deployed, neighbor nodes may be very close to each other. Hence, multihop communication in sensor networks is expected to consume less power than the traditional single hop communication. Furthermore, the transmission power levels can be kept low, which is highly desired in covert operations. Multihop communication can also effectively overcome some of the signal propagation effects experienced in long-distance wireless communication. The large number raises scalability issues on one hand, but provides a high level of redundancy on the other hand.

- Sensor nodes should be distributed for processing and sensing:

  In most cases, once deployed, WSNs have no human intervention. Hence the nodes themselves are responsible for reconfiguration in case of any changes. Using a wireless sensor network, many more data can be collected compared to just one sensor. Even deploying a sensor with great line of sight, it could have obstructions. Thus, distributed sensing provides robustness to environmental obstacles. Each sensor node should

be able to process local data, using filtering and data fusion algorithms to collect data from environment and aggregate this data, transforming it to information.

- The topology of a WSN changes very frequently:

  It is required that a sensor network system be adaptable to changing connectivity (for e.g., due to addition of more nodes, failure of nodes etc.) as well as changing environmental conditions. Thus, unlike traditional networks, where the focus is on maximizing channel throughput or minimizing node deployment, the major consideration in a sensor network is to extend the system lifetime as well as the system robustness.

- Sensor nodes are limited in power, computational capacities, and memory:

  Sensor nodes are small-scale devices with volumes approaching a cubic millimeter in the near future. Such small devices are very limited in the amount of energy they can store or harvest from the environment. Therefore there is only a finite source of energy, which must be optimally used for processing and communication. An interesting fact is that communication dominates processing in energy consumption. Thus, in order to make optimal use of energy, communication should be minimized as much as possible. Limited size and energy also typically means restricted resources (CPU performance, memory, wireless communication bandwidth and range).

- Sensor nodes may not have global identification (ID) because of the large amount of overhead and large number of sensors:

  Since the number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network, global identification (ID) is not proper for the Wireless Sensor Networks.

One of the most important constraints on WSNs is the low power consumption requirements. While traditional networks aim to achieve high quality of service (QoS) provisions, sensor network protocols must focus primarily on power conservation. They must have inbuilt trade-off mechanisms that give the end user the option of prolonging network lifetime at the cost of lower throughput or higher transmission delay.

1.2 Design principles for WSNs

In order to deal with the characteristics outlined above, some software design principles for WSN have already been proposed [2] [4]. In such papers, it has been evidenced the importance of localized algorithms.

Localized algorithms are distributed algorithms that achieve a global goal by communicating with nodes in some neighborhood only. Such algorithms scale well with increasing network size and are robust to network partitions and node failures. Adaptive fidelity algorithms allow to trade the quality of the result against resource usage and are thus a key element for resource efficiency. As an extreme case, the application can choose from a whole range of different algorithms which solve the same problem with different quality and resource requirements. Data-centric communication introduces a new style of node addressing by focusing on the data produced by nodes, since applications are unlikely to request the current sensor reading such as temperature at a specific node, but instead ask for locations where temperature exceeds a certain value. This allows for more robustness by decoupling data from the sensor that produced it. Finally, application knowledge in nodes can significantly improve the resource and energy efficiency, for example by application-specific data caching and aggregation in intermediate nodes.

However, severe energy constrains of sensor nodes require more energy efficient communication protocols over localized algorithm. The majority of the existing solutions are base on classical layered approach, e.g.,TCP/IP. Although design approach of classical layered protocol has many advantage, it is not best solution in terms of energy conservation in WSNs. It is more resource efficient approach to unify common protocol layer functionalities into a cross-layer module in WSNs. In fact, recent work on WSNs [6] shows that cross-layer integration is more energy efficient than classical layered protocol approach. Our design approach is based on cross-layer disciple that all functionalities of classical layered protocol are unified to a single protocol.

1.3 Overview on Routing protocol

One of the major issues in wireless sensor network is the design of energy-efficient routing protocols. Since sensor nodes have limited available power, energy conservation is a critical issue for nodes and network life in WSNs, as in section 1.1. Routing protocols in WSNs might differ depending on the application and network architecture. The Clear overview about routing protocols, with their constraints and design issues, has been proposed in [4].

This section discusses a taxonomy of routing protocols (see Fig. 1.1). The taxonomy shows how the routing protocols are categorized according to its protocol operation and network structure [4] [7].

The routing protocols for protocol operation are classified as follows:

- Multipath based routing

  These protocols offers fault tolerance by having at least one alternate path

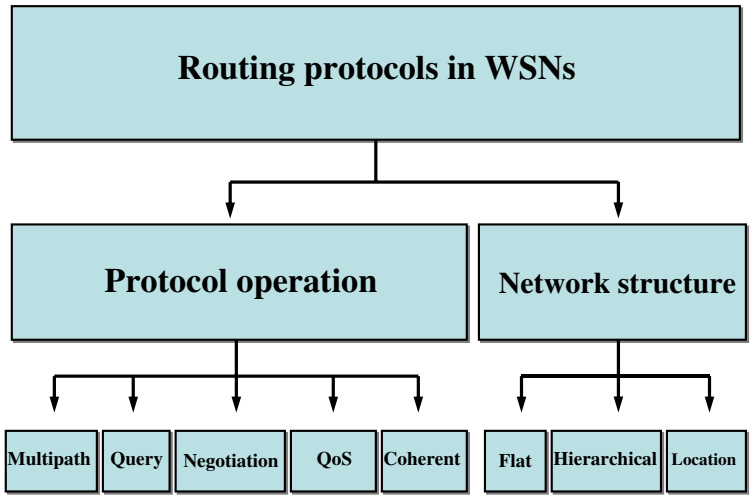

Figure 1.1: Taxonomy for Routing protocols

(from source to sink) and thus, increasing energy consumption and traffic generation. These paths are kept alive by sending periodic messages. The path with the largest residual energy when used to route data in a network may be very energy-expensive too, so there is a tradeoff between minimizing the total power consumed and the residual energy of the network, e.g., Directed diffusion (DD).

- Query based routing

  In this kind of routing, the destination nodes propagate a query for data (sensing task or interest) from the node through the network. The node containing this data sends it back to the node that has initiated the query, e.g., Rumor routing.

- Negotiation based routing

  These protocols use high-level data descriptors called "meta-data" in order to eliminate redundant data transmission through negotiations. The necessary decisions are based on available resources and local interactions, e.g., Sensor Protocols for Information via Negotiation (SPIN).

- QoS based routing

  In QoS-based routing protocols, the network has to balance between energy consumption and data quality. In particular, the network has to satisfy certain QoS metrics (delay, energy, bandwidth, etc.) when delivering data to the destination node, e.g., Sequential Assignment Routing (SAR).

- Coherent based routing

  In coherent routing, the data is forwarded to aggregators after minimum processing. The minimum processing typically includes tasks like timestamping and duplicate suppression. On the other hand, in noncoherent data processing routing, nodes will locally process the raw data before it is sent to other nodes for further processing, e.g., Single WinnEr algorithm (SWE).

The routing protocols for network structure are classified as follows:

- Flat-based routing

  In these protocols, all nodes have assigned equal roles in the network. Due to the large number of such nodes, it is not feasible to assign a global identifier to each node. This consideration has led to data-centric routing, where the destination node sends queries to certain regions and waits for data from the sensors located in the selected regions. Since data is being requested through queries, attribute-based naming is necessary to specify the properties of data, e.g., SPIN, DD, Rumor routing and Gradient-based routing (GBR).

- Hierarchical-based routing (Cluster-based routing)

  The nodes can play different roles in the network and normally the protocol includes the creation of clusters. The creation of clusters and assigning special tasks to cluster heads can greatly contribute to overall system scalability, lifetime, and energy efficiency. Hierarchical routing is an efficient way to lower energy consumption within a cluster, performing data aggregation and fusion in order to decrease the number of transmitted messages to the destination node. Additionally, designation of tasks for the sensor nodes with different characteristics are also preformed, e.g., Low Energy Adaptive Clustering Hierarchy (LEACH), Power-Efficient Gathering in Sensor Information Systems (PEGASIS), Threshold-Sensitive Energy Efficient Protocols (TEEN).

- Location-based routing

  In the protocols, the nodes are addressed by their location. Distance to next neighboring nodes can be estimated by signal strength or by GPS receivers, e.g., Geographic Adaptive Fidelity (GAF), Geographic and Energy Aware Routing (GEAR).

1.4 Overview of MAC protocols for WSNs

Medium access control (MAC) protocols have been developed to assist each node to decide when and how to access the channel. This problem is also

known as channel allocation or multiple access problem. The MAC layer is normally considered as a sublayer of the data link layer in the network protocol stack [8].

MAC protocols for WSNs must be energy efficient by reducing the potential energy wastes [2] [5]. We describes major sources of energy waste in MAC layer. When a node receives more than one packet at the same time, these packets are termed collided, even when they encounter partially. All packets that cause the collision have to be discarded and retransmissions of these packets are required, which increase the energy consumption. Although some packets could be recovered by a capture effect, a number of requirements have to be achieved for successful recovery. The second reason for energy waste is overhearing, meaning that a node receives packets that are destined to other nodes. The third energy waste occurs as a result of control-packet overhead. A minimal number of control packets should be used to make a data transmission. The fourth sources of energy waste is idle listening, that is, listening to an idle channel in order to receive possible traffic. The last reason for energy waste is overemitting, which is caused by the transmission of a message when the destination node is not ready. Given the above facts, a correctly designed MAC protocol should prevent these energy wastes.

To design a good MAC protocol for wireless sensor networks, the following attributes must be considered [9]. The first attribute is energy efficiency. We have to define energy-efficient protocols in order to prolong the network lifetime. Other important attributes are scalability and adaptability to changes. Changes in network size, node density, and topology should be handled rapidly and effectively for successful adaptation. Some of the reasons behind these network property changes are limited node lifetime, addition of new nodes to the network, and varying interference, which may alter the connectivity and hence the network topology. A good MAC protocol should gracefully accommodate such network changes. Other important attributes such as latency, throughput, and bandwidth utilization may be secondary in sensor networks. Contrary to other wireless networks, fairness among sensor nodes is not usually a design goal, since all sensor nodes share a common task.

Although there are various MAC layer protocols proposed for sensor networks, there is no protocol accepted as a standard. One of the reasons for this is that the MAC protocol choice will, in general, be application dependent, which means that there will not be one standard MAC for WSNs.

However, according to the underlying mechanism for collision avoidance, MAC protocols can be broadly divided into two groups in paper [10]: scheduled and contention-based.

1. Schedule-based protocols

   Schedule-based protocols are naturally energy preserving in that they

have a duty cycle built-in with an inherent collision-free nature, but they often have high complexity in design due to a non-trivial problem of synchronization in wireless sensor networks. We introduce the three schedule-based protocols: TDMA, FDMA and CDMA.

- TDMA

  TDMA has a natural advantage of collision free medium access. However, it includes clock drift problems and decreased throughput at low traffic loads due to idle slots. The difficulties with TDMA systems are synchronization of the nodes and adaptation to topology changes when these changes are caused by insertion of new nodes, exhaustion of battery capacities, broken links due to interference, the sleep schedules of relay nodes, and scheduling caused by clustering algorithms. The slot assignments, therefore, should be done with regard to such possibilities. However, it is not easy to change the slot assignment within a decentralized environment for traditional TDMA, since all nodes must agree on the slot assignments.

- FDMA

  FDMA is another scheme that offers a collision- free medium, but it requires additional circuitry to dynamically communicate with different radio channels. This increases the cost of the sensor nodes, which is contrary to the objective of sensor network systems.

- CDMA

  CDMA also offers a collision-free medium, but its high computational requirement is a major obstacle for the less energy consumption objective of sensor networks. In pursuit of low computational cost for wireless CDMA sensor networks, there has been limited effort to investigate source and modulation schemes, particularly signature waveforms, designing simple receiver models, and other signal synchronization problems. If it is shown that the high computational complexity of CDMA could be traded-off against its collision-avoidance feature, CDMA protocols could also be considered as candidate solutions for sensor networks.

2. Contention-based protocols

   Unlike scheduled protocols, contention protocols do not divide the channel into sub-channels or pre-allocate the channel for each node to use. Instead, a common channel is shared by all nodes and it is allocated on demand. A contention mechanism is employed to decide which node has the right to access the channel at any moment.

Contention protocols have several advantages compared to scheduled protocols. First, because contention protocols allocate resources on demand, they can scale more easily across changes in node density or traffic load. Second, contention protocols can be more flexible as topologies change. There is no requirement to form communication clusters, and peer-to-peer communication is directly supported. Finally, contention protocols do not require fine-grained time synchronizations as in TDMA protocols.

The major disadvantage of a contention protocol is its inefficient usage of energy. Nodes listen at all times and collisions and contention for the media can waste energy. Overcoming this disadvantage is required if contention-based protocols are to be applied to long-lived sensor networks.

- CSMA

  In accordance with common networking lore, CSMA methods have a lower delay and promising throughput potential at lower traffic loads, which generally happens to be the case in wireless sensor networks. However, additional collision avoidance or collision detection methods should be employed.

- ALOHA

  In ALOHA, a node simply transmits a packet when it is generated (pure ALOHA) or at the next available slot (slotted ALOHA). Packets that collide are discarded and will be retransmitted later.

A wide range of MAC protocols defined for WSNs are described briefly by stating the essential behavior of the protocols wherever possible. Moreover, the advantages and disadvantages of these protocols are presented.

- Sensor MAC (S-MAC)

  Locally managed synchronizations and periodic sleep–listen schedules based on these synchronizations form the basic idea behind the Sensor-MAC (S-MAC) protocol [9]. Building on contention-based protocols like 802.11, S-MAC strives to retain the flexibility of contention-based protocols while improving energy efficiency in multi-hop networks. S-MAC includes approaches to reduce energy consumption from all the major sources of energy waste: idle listening, collision, overhearing and control overhead. Neighboring nodes form virtual clusters so as to set up a common sleep schedule. If two neighboring nodes reside in two different virtual clusters, they wake up at the listen periods of both clusters.

  Schedule exchanges are accomplished by periodic SYNC packet broadcasts to immediate neighbors. The period for each node to send a SYNC

packet is called the synchronization period. Collision avoidance is achieved by a carrier sense. Furthermore, RTS/CTS packet exchanges are used for unicast-type data packets.

Periodic sleep may result in high latency, especially for multihop routing algorithms, since all intermediate nodes have their own sleep schedules. The latency caused by periodic sleeping is called sleep delay. The adaptive listening technique is proposed to improve the sleep delay and thus the overall latency. In that technique, the node that overhears its neighbor's transmissions wakes up for a short time at the end of the transmission. Hence, if the node is the next-hop node, its neighbor could pass data immediately. The end of the transmissions is known by the duration field of the RTS/CTS packets.

The energy waste caused by idle listening is reduced by sleep schedules in S-MAC. In addition to its implementation simplicity, time synchronization overhead may be prevented by sleep schedule announcements.

However broadcast data packets do not use RTS/CTS, which increases collision probability. Adaptive listening incurs overhearing or idle listening if the packet is not destined to the listening node. Sleep and listen periods are predefined and constant, which decreases the efficiency of the algorithm under variable traffic load.

- Timeout MAC (T-MAC)

  The static sleep–listen periods of S-MAC result in high latency and lower throughput, as indicated above. Timeout-MAC (T-MAC) [11] is proposed to enhance the poor results of the S-MAC protocol under variable traffic loads. In T-MAC, the listen period ends when no activation event has occurred for a time threshold TA. The decision for TA is presented along with some solutions to the early sleeping problem defined in [11]. Variable loads in sensor networks are expected, since the nodes that are closer to the sink must relay more traffic and traffic may change over time. Although T-MAC gives better results under these variable loads, the synchronization of the listen periods within virtual clusters is broken. This is one of the reasons for the early sleeping problem.

- Berkeley MAC (B-MAC)

  B-MAC is highly configurable and can be implemented with a small code and memory size [12]. It has an interface that allows choosing various functionality and only that functionality as needed by a particular application. B-MAC consists of four main parts: clear channel assessment (CCA), packet backoff, link layer acks, and low power listening. For CCA, B-MAC uses a weighted moving average of samples when the

channel is idle in order to assess the background noise and better be able to detect valid packets and collisions. The packet backoff time is configurable and is chosen from a linear range as opposed to an exponential backoff scheme typically used in other distributed systems. This reduces delay and works because of the typical communication patterns found in a wireless sensor network. B-MAC also supports a packet by packet link layer acknowledgement. In this way only important packets need pay the extra cost. A low power listening scheme is employed where a node cycles between awake and sleep cycles. While awake it listens for a long enough preamble to assess if it needs to stay awake or can return to sleep mode. This scheme saves significant amounts of energy. Many MAC protocols use a request to send (RTS) and clear to send (CTS) style of interaction. This works well for ad hoc mesh networks where packet sizes are large (1000s of bytes). However, the overhead of RTS-CTS packets to set up a packet transmission is not acceptable in wireless sensor networks where packet sizes are on the order of 50 bytes. B-MAC, therefore, does not use a RTS-CTS scheme.

- Zebra MAC (Z-MAC)

  Z-MAC is a hybrid MAC scheme for sensor networks that combines the strengths of TDMA and CSMA while offsetting their weaknesses [13]. The main feature of Z-MAC is its adaptability to the level of contention in the network so that under low contention, it behaves like CSMA, and under high contention, like TDMA. By mixing CSMA and TDMA, Z-MAC becomes more robust to timing failures, time-varying channel conditions, slot assignment failures and topology changes than a stand-alone TDMA.

There are various MAC protocols for WSNs besides the protocols we here presented above. Optimal choice of MAC protocols is determined by application specified goals such as accuracy, latency, and energy efficiency. However B-MAC protocol is widely used because has good results even with default parameters and performs better than the other studied protocols in most cases.

Chapter 2

Randomized Protocol

In this chapter, we investigate the problem of maximizing the network lifetime under the reliability and stability constraints specified by given application using Randomized Protocol. The extended points of Randomized Protocol will be explained in Chapter 3.

Basically, the constraints are listed as:

1. Error rate guarantee;
2. End-to-End (E2E) delay guarantee;

Furthermore, the cost function is derived from the energy consumption in network.

We also study a completely distributed adaptation algorithm that allows the network to reach the optimal working point by adapting the traffic and channel variations without high overhead. Therefore in the tradeoff between energy expenditure and reliability the overall system efficiency should be maximized in terms of energy consumption.

First of all, we will start with a brief description of the Randomized Protocol in section 2.1. Problem definition will follow in Section 2.2. In Section 2.3 we will give a brief description about the protocol stack, and a mathematical formulation to derive the optimization problem will be analyzed in Section 2.4. The mathematical formulation allows the computation of the system efficiency for a specific way of choosing the system parameters.

2.1 Introduction to the Randomized Protocol

Wireless sensor networks, in most applications, are required to have a long lifetime in the order of months to years while the constituent sensor nodes have limited battery power. Since saving energy is the most important goal

in a wireless sensor networks, it is used as the main optimization objective, while other objectives as throughput, delay and reliability are less important. But shortsighted optimization for energy can lead to sensor networks that can not fulfill their tasks. Hence, energy efficiency must be balanced with the requirements of the tasks that are performed by WSNs. Decaying costs of sensor nodes will allow to deploy high densities and we believe that leveraging this resource is the key to ensure reliable communication out of random behavior as nodes malfunctioning and failure, harsh communication performance and robustness. Therefore protocol design should adapt to the inherent randomness of WSNs systems using high node densities. We use the Randomized Protocol [14] which is a novel cross-layer framework for designing WSNs. Main idea of this protocol is adopting a cross-layer strategy combining the randomized routing, MAC and adaptive sleeping discipline to support the wireless automation. In the paper [14], the authors investigate the problem of the lifetime maximization in a WSN under the constraints of the target end-to-end delay and error rate. Furthermore, the authors demonstrate how this cross-layer can work for energy efficiency while guaranteing constraints on the optimal working point.

2.2 Problem Definition

Saving energy is the most important goal in a sensor network. In our scenario, there are several sources sending packets to destination node with the traffic rate of λ in the source block. From source to destination a high density of intermediate nodes is uniformly deployed to relay the packets generated by the sources. The communication network should provide the following services:

1. Error Rate guarantee

   Packet Reception Rate (PRR) defined as the probability that a packet is received at destination. It should be higher than Ω:

   $P[correct] \geq \Omega$

2. Delay guarantee

   The delay of packet delivery from sources to destination should be constrained within τ seconds on required probability:

   $P[E2E \leq \tau] \geq 0.96$

The two predefined constraints can be used to express an optimization problem where the cost function is the energy consumption.

There are a number of assumptions to solve the constrained optimization problem:

- Each node knows its location and sources and destination;
- Each node knows total number of deployed nodes and network size (i.e., the distance from source to destination);
- There are high density of nodes with tunable transmitting power between source block and destination.

The main ideas of this protocol can be summarized as follows:

1. The selection of the next hop is a random choice among nodes of a calculated region (i.e., next cluster).
2. The adaptive sleeping algorithm is applied to nodes.
3. Random contention scheme prevents the extra transmit energy consumption from duplicate packets.
4. Fixed channel allocation (FCA) decreases collision probability by reducing interference between beacons and data messages.

In the next section, these characteristics will be explained and discussed.

2.3 Protocol Design

2.3.1 Cross-Layer Solution

The Extended Randomized Protocol is combined with the randomized routing, randomized MAC, contention scheme, frequency division access scheme and adaptive sleeping disciple.

- Randomized routing

  There are many challenges and design issues that affect the routing process in WSNs (see Section 1.1). However, the main challenges can be summarized with: random node deployment, energy consumption without losing accuracy, fault tolerance from failed sensor nodes due to lack of power, physical damage, or environmental interference. Routing over an unpredictable environment and energy constraint is notoriously hard. High node density makes the problem easier to solve. In addition, the overhead of routing process should be minimized to save the energy.

  The basic idea of the Randomized Protocol is to have a set of nodes within transmission range that could be candidate receivers with low overhead. Randomized routing makes a route to transmit packets without knowledge of the next hop neighborhood. The sender has knowledge of the transmission region to which the packet will be forwarded, but the actual

choice of forwarding node is made at random. Consequently, Randomized Protocol saves the energy due to unnecessary node coordination, state maintenance of neighbor nodes, and increases the robustness to nodes failures.

- Adaptive sleeping disciple

  The most important way to save energy in a sensor network is to power-down (put to sleep) any node that is not performing useful work. Therefore most of sleeping disciplines try to put the nodes to sleep while preserving a connectivity graph and rely upon strong synchronization in the network. Because sensor nodes are often densely deployed (i.e., to support high-resolution sampling of the environment), there exists a high degree of redundancy in the network topology. Thus, it is possible to design a node communication protocol that can exploit this redundancy and allow nodes to minimize energy consumption by sleeping for the maximum amount of time.

  We use a lightweight, distributed adaptive sleeping discipline [15] that meets these goals while ensuring overall network performance requirements (e.g., routing delay) are met. Another important aspect of the adaptive sleeping discipline is robustness and adaptation to changing network connectivity. The network topology is time-varying due to the addition of new nodes (birth), energy depletion in others (death), and the mobility of nodes. Furthermore, real-world deployment of sensor networks has revealed that, even without birth, death, and mobility, sensor network connectivity varies overtime. The adaptive sleeping disciple ensures robustness to changing network connectivity and provisions for an adaptive scheme that performs well under various network conditions. According to adaptive sleeping discipline in the improved Randomied protocol, each node goes to sleep for an amount of time that is a random variable whose parameters are a function of traffic rate and network channel condition.

- Contention scheme

  In Randomized Protocol, there is no mechanism to prevent the duplicate packets increasing the traffic load and energy consumption in network [16]. Transmit nodes implement a random contention scheme [17] to discard duplicate packets instead of directly sending a packet.

- Frequency division access scheme

  In frequency division access scheme the given Radio Frequency (RF) bandwidth is divided into smaller frequency bands. To avoid interference between beacons and data messages in a channel, we allocate the different

channel for beaconing and data communication using the fixed channel allocation (FCA).

2.3.2 State Machine

A wireless sensor network is characterized as a massively distributed and deeply embedded system [18]. Such a system requires concurrent and asynchronous event handling as a distributed system and resource-consciousness as an embedded system. State machine based software design techniques are capable of satisfying exactly these requirements.

In this section, we study a state machine to design a compact and efficient protocol for WSN. Consider a node of network. The behavior of a node can be explained my considering the state machine of Fig. 2.1. In practice, a node can stay in six states:

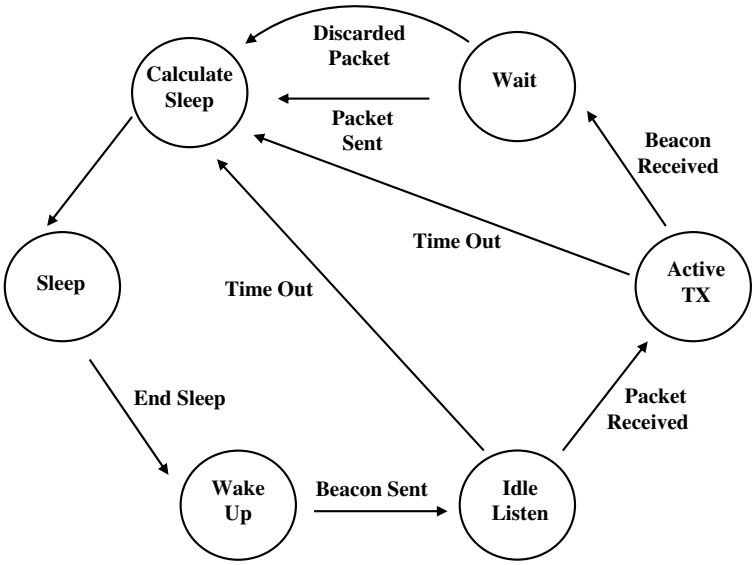

Figure 2.1: State diagram

SLEEP STATE: The node turns off its radio and keeps Low Power Mode (LPM) and starts a grenade timer whose duration is an exponentially distributed random variable of intensity μ_i. When the timer expires, the node goes to the WAKE UP STATE.

WAKE UP STATE: The node turns its beacon channel on and wakes up from LPM and broadcasts a beacon message containing the channel condition p. After the node sends a beacon message, its radio is converted to the data channel and it is ready to receive a data message. The node goes to the IDLE LISTEN STATE.

IDLE LISTEN STATE: The node starts a grenade timer of a fixed active time T_ac that must be long enough to completely receive a packet. If a data message is received, the active timer is discarded and the node starts a wait timer T_w to receive a beacon that is a longer time than the active time. The node renews the traffic rate λ and intensity parameter μ_i from the received data message and goes to the ACTIVE TX state. Furthermore its radio is switched from the data channel to the beacon channel. Otherwise if the active timer expires before any packet is received, the node goes to the CALCULATE STATE.

ACTIVE TX STATE: After a node receives the first beacon coming from a node in the next cluster within wait timer T_w, it calculates the cluster ID and the size of next cluster using received beacon message containing the channel condition p and data message containing the traffic rate λ in network. Next cluster is the region that has nodes which generate a beacon to receive data message. After the node computes the next cluster size, node goes to WAIT STATE instead of directly transmitting the data message. Otherwise if the beacon waiting timer expires before receiving any beacon message, the node goes to the CALCULATE STATE.

WAIT STATE: Node starts a backoff time T_d before transmitting a data message and its radio is switched to the data channel. The back-off time T_d is a uniformly distributed random variable within 0 to a maximum value called T_{dmax}. If the node listens a data message whose sequence number is same with own data message within a backoff time T_d, the node discards the own data message and goes to CALCULATE STATE to avoid duplicate packets. Otherwise if the backoff timer T_d expires, the node transmits the data message and goes to CALCULATE STATE.

CALCULATE STATE: The node calculates the next sleeping time from the intensity parameter μ_i and generates an exponentially distributed random variable of mean $1/\mu_i$. After the node generates the random variable for sleeping time, the node goes back to the SLEEP STATE.

2.4 Mathematical Model

From the protocol design, there are two critical parameters, i.e. the wake up rate μ_i of node and size of next cluster. To estimate these two critical parameters, we model the network as a constrained optimization problem, where the constraints are the E2E delay and error rate requirements and the cost function is the energy consumption of network.Consider μ_i the wake up rate of node i and $\mu_{c,i}$ the cumulative wake up rate of cluster i. Although node requires optimal wake up rate of each node to control μ_i, our algorithm will provide μ_o the

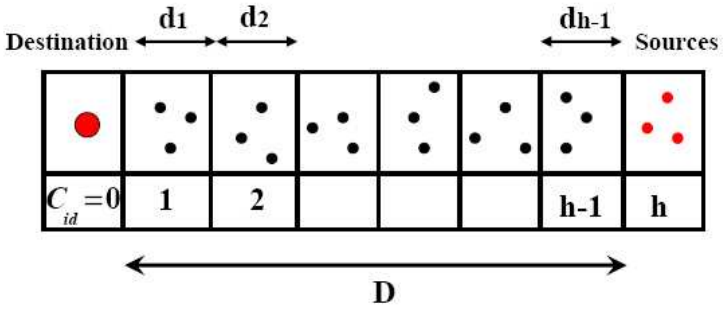

Figure 2.2: Block abstraction

optimal wake up rate of each block. However, this problem will be solved in Section 2.6. In the Fig. 2.2, we introduce the block abstraction to build a simplified mathematical model of the protocol. In this block abstraction, we divide a node layout in $h-1$ clusters corresponding to forwarding regions. For example, nodes in cluster i can forward packets only to a node in cluster $i-1$ without interference. Where the C_{id} is a cluster ID for each block. Consequently, important parameters to optimize energy consumption are the number of clusters $h-1$ and size of each cluster, together with the cumulative wake up rate $\mu_{c,i}$ of the nodes within each cluster.

2.4.1 First Order Simple Circuit Model

There are different assumptions about the radio characteristics, including energy dissipation in the transmit and receive modes.

In our investigation, we assume a first order circuit model to derive the energy consumption for the different modes on sensor nodes [19]. Since we use the Tmote Sky where the node dissipates $E_{Te} = 234.0nJ/bit$ to run the transmitter circuitry, $E_{Re} = 261.6nJ/bit$ to run the receiver circuitry (see Figure 2.3 and Table 2.1) from datasheet [20].

Table 2.1: Node energy specification

Operation	Energy Consumption
E_{Te} = Power consumption on Tx mode / R	$234.0nJ/bit$
E_{Re} = Power consumption on Rx mode / R	$261.6nJ/bit$

where R is the transmission rate $250kbps$ on Tmote Sky.

The energy consumption to transmit a l bit message at a distance d due

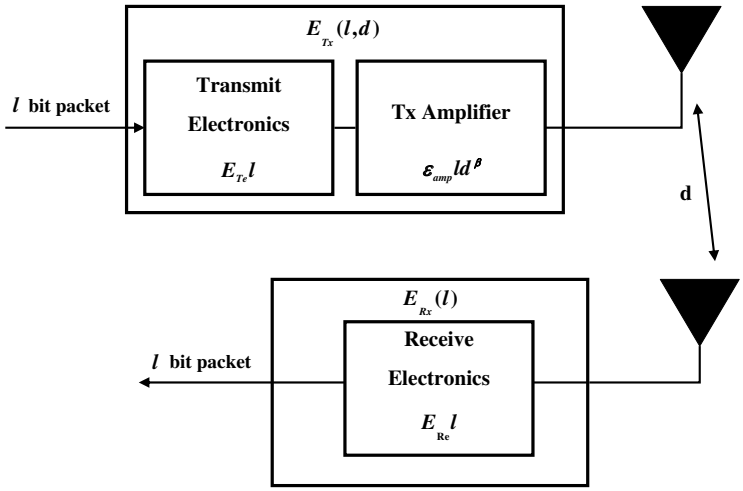

Figure 2.3: First order circuit model of the transmitter and receiver

to the channel transmission is derived from the power control algorithm in Section 3.3:

$$\epsilon_{amp} = \frac{\overline{\gamma}_{con}\ PL(d_0)\ P_{ni}\ 10^{\frac{log_e(10)}{200}}\ \sigma^2}{R} \tag{2.1}$$

where the constraint on the average SINR is $\overline{\gamma}_{con} \geq \frac{\left(2P_{con}^{1/l}-1\right)^2}{1-\left(2P_{con}^{1/l}-1\right)^2}$, the path loss at the reference distance is $PL(d_0)$, the noise floor and interference is P_{ni} and σ denotes the standard deviation of a Gaussian attenuation.

Thus, to transmit a l bit message at a distance d using our simple circuit model, the node consumes:

$$E_{Tx}(l,d) = E_{Te}\ l + \epsilon_{amp}\ l\ d^{\beta} \tag{2.2}$$

where β is the path loss exponent.

To receive a message having the same size as before, the radio uses:

$$E_{Rx}(l) = E_{Re}\ l \tag{2.3}$$

Note that, with these value of the parameters, receiving a message is not a low cost operation. Furthermore, listening mode consumes similar amount of energy as the receiving mode on Tmote Sky from datasheet [20]. Therefore, the protocols should minimize not only the transmit distances but also the num-

ber of transmit, receive and listening operations for communication of each message.

2.4.2 Constraints

In this section, we describe the two problem constraints using a mathematical model.

1. Error Rate guarantee

   Since we do not implement the Automatic Repeat reQuest (ARQ), a packet can be lost at each hop because of a collision or a bad channel during transmission.

   (a) Bad channel

   To obtain simple channel modelling, Bernoulli model is applied with the probability of having a good channel p during a single transmission.

   (b) Collisions

   Considering the case of a node in cluster i send a data message to a node in cluster $i-1$, a collision occurs if another node in cluster i receives a data message before a node in cluster $i-1$ has broadcasted a beacon message to node in cluster i. The collision probability is caused by the incoming traffic of cluster i. It can be determined considering a Poisson process of intensity λ and cumulative wake up rate $\mu_{c,i}$ in cluster i. Thus the collision probability in cluster i is $P[coll] = \frac{\lambda}{\lambda+\mu_{c,i}}$ using a Bernoulli model channel model.

   Consequently, probability of successful packet transmission is given by $P_{s,i} = \frac{p\mu_{c,i}}{\lambda+\mu_{c,i}}$ in cluster i. Assuming h hops in network, the error constraint becomes:

$$\prod_{i=1}^{h} \frac{p\mu_{c,i}}{\lambda + \mu_{c,i}} \geq \Omega \tag{2.4}$$

2. End-to-End Delay guarantee

   The E2E delay between source and destination is given by the sum of the delays at each hop. There are three sources of delay at each hop.

   (a) Time to wait before sending a data message:

   Since the wake up rate of each node is an exponentially distributed random variable, the time to wait the beacon message before the first wake up in the next cluster happens also an exponentially distributed random variable. Therefore intensity of the wake up rate

of the next cluster is the sum of wake up rates of each nodes in the next cluster.

(b) Time to forward a packet once the connection is established:

Constant F is the time containing propagation delay and transmission time of a data message. Therefore constant F is depend on a data size l_m and transmission rate R.

(c) Random delay time to avoid duplicate packets:

After node receives a beacon message, node starts a backoff time T_d instead of directly sending data message. The backoff time T_d is a uniformly distributed random variable within 0 to a maximum value called T_{dmax}.

Assuming h hops, the E2E delay constraint in the worst case contention scheme delay becomes:

$$P[\sum_{i=1}^{h} \alpha_i + hF + (h-1)T_{dmax} \leq \tau] \geq 0.96 \tag{2.5}$$

where α_i is an exponentially distributed random variable having parameter $\mu_{c,i}$.

2.4.3 Cost Function

The energy consumption results from the transmission and reception of data messages, wake up rate, wait and beaconing in network. In our analysis, we divide total energy consumption in network into two parts that spend on transmission and reception, and that spend on active time including wake up and beaconing.

1. Transmission and Reception :

   From the Section 2.4.1, we use the first order simple circuit model to derive the energy consumption for the transmission and reception of a data message. In addition, the random contention scheme to prevent duplicate packets consumes $E_{delay} = P_{Re} \frac{T_{dmax}}{2}$ where the P_{Re} is the receiving mode power consumption and mean time of random contention scheme $\frac{T_{dmax}}{2}$. The energy consumption to transmit a l_m bit data message at a distance d using our simple circuit model is:

$$E_{Tx}(l_m, d) = E_{delay} + E_{Te}\, l_m + \epsilon_m\, l_m\, d^{\beta} \tag{2.6}$$

   where the E_{Te} is the unit energy consumption on TX mode per bit due to the RF circuit, β is the path loss exponent $2 \leq \beta \leq 6$.

$$\epsilon_m = \frac{\overline{\gamma}_{cot_m}\, PL(d_0)\, P_{ni}\, 10^{\frac{log_e(10)}{200}\sigma^2}}{R} \tag{2.7}$$

Consider the log-normal shadowing model from Section 2.4.1 and $\overline{\gamma}_{con} \geq \frac{\left(2P_{con}^{1/l_m}\right)^2}{1-\left(2P_{con}^{1/l_m}\right)^2}$.

to receive same size of message, the radio uses:

$$E_{Rx}(l_m) = E_{Re}\, l_m \tag{2.8}$$

where E_{Re} is the unit energy consumption on RX mode per bit due to the RF circuit.

Let's assume that the source emits $T\lambda$ packets during the time T with h hops in network, the energy consumption for transmission and reception associated to correctly received these data message becomes:

$$E_{pck} = T\lambda \sum_{i=1}^{h} (E_{delay} + E_{Te}\, l_m + \epsilon_m\, l_m\, d_i^{\beta} + E_{Re}\, l_m) \tag{2.9}$$

2. Wake up and Beaconing:

   Each time a node wakes up, it contributes a fixed RX mode energy consumption:

$$E_{ac} = P_{Re}\, T_{ac} \tag{2.10}$$

   where the P_{Re} is the receiving mode power consumption and T_{ac} is fixed active time that must be long enough to completely receive a data message. Consider the channel condition p, nodes have to wake up on average $1/p$ times to create the effect of a single wake up. Assuming h hops and a cumulative wake up rate per cluster $\mu_{c,i}$, the total cost for wake ups and transmit a l_b bit beacon message at a distance d:

$$E_{WU} = \frac{1}{p} \sum_{i=1}^{h} \mu_i (E_{ac} + \epsilon_b\, l_b\, d_i^{\beta}) \tag{2.11}$$

   where the ϵ_b is same with Eq. 2.7 except instead of l_m there is the beacon size l_b and β is the path loss exponent.

   After we sum the two parts, the total energy consumption of the network becomes:

$$E_{tot} = T\Big(\lambda\big(h\, E_{delay} + h\, E_{Re}\, l_m + h\, E_{Te}\, l_m + \sum_{i=1}^{h} \epsilon_m\, l_m\, d_i^{\beta}\big) + \frac{1}{p}\mu_i\big(hE_{ac} + \sum_{i=1}^{h} \epsilon_b\, l_b\, d_i^{\beta}\big)\Big) \quad (2.12)$$

Although some of the packets are lost for collisions, in Eq. 2.12 we implicitly assumed all the packets getting to destination. Thus Eq. 2.12 gives a upper bound on the energy consumption.

As it will be clearer in Section 2.5.1, this approximation allows a manageable solution of the optimization problem.

Consequently, the constrained optimization problem becomes:

$$Arg\ min_{(h,d_1,\ldots,d_h,\mu_{c,1},\ldots,\mu_{c,h})} E_{tot} \quad (2.13)$$

constrained by error rate inequality 2.4 and E2E delay inequality 2.5.

2.5 Optimization Algorithm

We study the algorithm to find the optimal size of each block and the number of clusters, the optimal wake up intensity μ_o. In the Section 2.5.1, we deal with problem reduction to derive the optimization algorithm. The Section 2.5.2 proposes the optimization algorithm to derive an optimal working points.

2.5.1 Problem Reduction

Considering the transmission energy term for a data and beacon message in Eq. 2.12, the transmission energy terms is combined with the sum of positive power for cluster size d_i. Since the path loss exponent $\beta \geq 2$ and the block size d_i are strictly positive, in Eq. 2.12 the same size of each clusters d_i satisfies the minimum cost to transmit data and beacon message. That is the size of cluster becomes $d_1 = d_2 = \ldots = d_h = D/h$ for end to end distance D and any strictly positive integer, number of hops h.

Furthermore, let's assume that each block has the same collision probability and channel condition, this generates the same incoming traffic load for each clusters. Therefore a cluster with a lower cumulative wake up rate would create a bottleneck link in the communication infrastructure. Consequently, the cumulative wake up rate should be the same for every cluster $\mu_{c,1} = \mu_{c,2} = \ldots = \mu_{c,h} = \mu_c(h)$ to avoid the bottleneck link in the communication infrastructure.We solve the optimization problem from these observations.

1. E2E delay constraint:

   Consider the E2E delay constraint equation 2.5 and call $A(h) = \sum_{i=1}^{h} \alpha_i$. Since the $\alpha's$ are i.i.d., it can apply the central limit theorem and approximate $A(h)$ with a Gaussian random variable. That is

   $$A(h) \in N\big(\frac{h}{\mu_c(h)}, \frac{h}{(\mu_c(h))^2}\big).$$

   Consequently, the E2E delay constraint becomes:

   $$\mu_c(h) \geq \frac{h + 2\sqrt{h}}{\tau - h(F + T_{dmax}) + T_{dmax}} \tag{2.14}$$

   Call $D_c(h) = \frac{h+2\sqrt{h}}{\tau - h(F+T_{dmax})+T_{dmax}}$. Thus inequality 2.14 introduces the constraint for total number of blocks $h \leq \frac{\tau + T_{dmax}}{F + T_{dmax}}$ from the denominator of $D_c(h)$.

2. Error rate constraint:

   The error rate constraint becomes $(\frac{p\mu_c(h)}{\mu_c(h)+\lambda})^h \geq \omega$. The constraint on the wake up rate can be expressed as $\mu_c(h) \geq \frac{\lambda\omega^{1/h}}{p - \omega^{1/h}}$. Since sensor node uses a microcontroller, the computation should be minimized in contrast to a general-purpose microprocessor. Using Taylor expansion on h, the constraint on the wake up rate can be approximated as:

   $$\mu_c(h) \geq \frac{\lambda h}{h\, ln(p) - ln(\Omega)} \tag{2.15}$$

   Call $E_c(h) = \frac{\lambda h}{h\, ln(p) - ln(\Omega)}$. Note that inequality 2.15 introduces the constraint $h \leq \frac{ln(\Omega)}{ln(p)}$ from denominator of $E_c(h)$.

3. Cost function:

   Consider the condition $D_c(h)$, $E_c(h)$ and reorganize the Eq. 2.12. Consequently, the constrained optimization problem becomes:

   $$Arg\ min_h\ T\Big(\lambda\big(h\, E_{delay} + h\, E_{Re}\, l_m + h\, E_{Te}\, l_m + \epsilon_m\, l_m\, D^{\beta} h^{1-\beta}\big) + \frac{max\,\{D_c(h), E_c(h)\}}{p}\big(hE_{ac} + \epsilon_b\, l_b\, D^{\beta} h^{1-\beta}\big)\Big) \tag{2.16}$$

   where $max\,\{D_c(h), E_c(h)\}$ becomes the optimal cumulative wake up rate μ_o.

   In Eq. 2.16, $h^{1-\beta}$ is convex function for $\beta \geq 2$. Since the equation 2.16 is convex combination, the optimization problem is convex function within

$$0 \leq h \leq min\left\{\frac{\tau + T_{dmax}}{F + T_{dmax}}, \frac{ln(\Omega)}{ln(p)}\right\}.$$

2.5.2 Algorithm

Although the total energy consumption E_{tot} is a convex function, it is in general non differentiable and finding a closed solution is not always possible. Since optimal integer value of hops h exist for

$$0 \leq h \leq min\left\{\frac{\tau + T_{dmax}}{F + T_{dmax}}, \frac{ln(\Omega)}{ln(p)}\right\},$$

a simple iterative algorithm can be applied to find the optimal number of blocks h.

Initialize: **Evaluate** $Res = Etot(1)$, set $h = 2$
Step: if $\Big(\big(Etot(h) < Res\big)$ && $\big(h \leq min\left\{\frac{\tau+T_{dmax}}{F+T_{dmax}}, \frac{ln(\Omega)}{ln(p)}\right\}\big) \Big)$
$Res = E_{tot}$;
h++;
Go to **Step**;
else
Return h- -, $\mu_o = max\left\{D_c(h--), E_c(h--)\right\}$
end;

In this iterative algorithm,it can be proved that the worst case number of iterations is $min\left\{\frac{\tau+T_{dmax}}{F+T_{dmax}}, \frac{ln(\Omega)}{ln(p)}\right\} - 1$ times.

2.6 Distributed Adaptation Protocol

In the previous Section, we studied how to determine the optimal cluster size D/h and cumulative wake up rate μ_o in network. In this Section, we report on a distributed algorithm for self-organizing sensor networks that respond to variation of the traffic rate λ of the application and channel condition p. Each node has to run correctly to determine its forwarding region and wake up rate so that the overall network operates at the optimal working point calculated in 2.5.2. In addition, the proposed distributed protocol works locally with low message overhead.

2.6.1 Preliminaries

Node needs to know the traffic rate λ and channel condition p to solve the constrained optimization problem in Eq. 2.16. However these quantities can not be locally estimated in each nodes. We add the traffic rate λ that can piggybacked on a data message from the source block. Furthermore, if the data messages are numbered, the destination node can easily estimate the channel

condition p and the value can be piggybacked on beacon messages. Thus each node recognizes the change in traffic rate λ from the data message and change in channel condition p from the beacon message with small message overhead.

Assume that the total number of nodes is N, and n is the number of nodes in a cluster. Ideally, we favor the solution of distributing the cumulative wake up rate equally between all the nodes. Calling μ_i the wake up rate of node i, the fair optimal solution is $\mu_i = \frac{\mu_o}{n}\ \forall i = 1, \ldots, n$. However, a node does not know and cannot efficiently estimate the number of nodes in its block. Furthermore, even if we assume that nodes know the total number of nodes N that are deployed in the sensor field, nodes can not estimate the number of nodes $n = N/h$ in a block because of node mobility, node failures due to depleted batteries or environmental influences, and so on. This problem was addressed in [15]. In that paper, a parallel was drawn between this problem and the fair bandwidth allocation for TCP flows and it was shown how implementing an Additive Increase and Multiplicative Decrease (AIMD) algorithm of the wake up rate of each node leads to a fair distribution of the wake up duties within a single block. We follow this adaptive sleep discipline approach. Specifically, each node that is waiting to forward a data message observes the time before the first wake up in the next cluster. Starting from this observation, it estimates the cumulative wake up rate of the next cluster and it compares it with optimal cluster wake up rate μ_o calculated through the iterative algorithm outlined in the previous section. If the estimated value is less than or equal to the optimal value, it communicates to the next hop to additively increase its wake up rate, otherwise it orders the next hop to multiplicatively decrease its wake up rate. Furthermore the exponential filter is used to estimate the next cluster wake up rate. Calling α the time observed before the first wake up in the next cluster, the new estimated wake up rate is $\mu_{new} = b\mu_{old} + \frac{1-b}{\alpha}$. We use a $b = 0.6$ from the simulation result [14]. The AIMD command on the wake up rate variation is piggybacked on the data message and it does not require any additional transmission.

In the Randomized protocol, node calculates the size of the forwarding region (FWR). The FWR is the region between the maximum and minimum distance (d_{max}, d_{min}) at which the next hop must be. However, since a network environment in sensor network has higher channel condition variation and there is a strict limitation for power control in sensor node, it is hard to estimate the accurate radio propagation distance using simple power control algorithm. A simple clustering technique is described to know the next cluster region and to eliminate duplicate packet problem. Furthermore, clustering technique is particularly useful for applications that require scalability to hundreds or thousands of nodes [21]. Scalability in this context implies the need for load balancing and efficient resource utilization. The essential operation in

sensor node clustering is to select a set of cluster heads among the nodes in the network, and cluster the rest of the nodes with these heads. However Randomized protocol does not need to select a cluster heads in each blocks. Each node independently makes its decisions based on local information in our simple clustering technique. We assume that the network is pre-configured, i.e. each sensor has an cluster ID based on location in network. However, we do not require time synchronization. After nodes receive the change in traffic rate or channel condition, nodes run the Iterative Algorithm in Section 2.5.2 and determine the optimal number of clusters h.

Assume N nodes in a network. Therefore each node retrieves a cluster ID using simple clustering technique that divides total number N nodes to h number of clusters based on location information. For example, consider the 12 nodes and optimal number of clusters $h = 2$ in network. Simple cluster technique assigns a cluster ID 1 from node number 1 to 6 closer to a destination node, a cluster ID 2 from node number 7 to 12 closer to a source block.

2.6.2 Distributed Protocol

1. destination Node

   - **INIT STATE:** The node sets its transmission power that can send beacon to distance D, wake up rate $\mu_i = \mu_{init}$, channel condition $p = \Omega + \delta$ and cluster ID $C_{id} = 0$, where the D is network size and δ is a small constant term. Thus a initial channel condition p is slightly higher than required packet reception rate Ω. If the channel condition p is lower than Ω, then each node maximizes the wake up rate in Eq. 2.15, i.e. the network consumes the maximum energy. When the destination node receives data messages more than a threshold Θ_{seq}, $\Theta_{seq} = \Omega\,\Theta_{max,s}$, it estimates the first channel condition and sends it on a beacon message to cluster ID $C_{id} = 1$. Since the cumulative PRR will increase over the time in initial state, the first estimation time of channel condition should depend on Ω. Furthermore the destination node starts a periodic timer whose period $T_{ch,p} = \frac{\psi_{max}}{\lambda\,\Omega}$ where ψ_{max} is maximum time to estimate a channel condition. When the timer expires, destination node goes to the **REP STATE**. Since the received number of packet in destination node is depend on the traffic rate λ and constraint of error rate Ω, period to estimate channel condition should depend on two parameters.
   - **REP STATE:** After the periodic timer expires, destination node estimates new channel condition using a total number of received data messages in a period $T_{ch,p}$. If there is a change in the channel condition p, the destination node runs the Iterative Algorithm and de-

termines a new optimal number of clusters h, transmission power on distance D/h to send beacon message. If there is a change in traffic rate λ on a data message, the destination node executes the Iterative Algorithm and determines the optimal number of clusters h, transmission power on distance D/h to send beacon message.

2. Intermediate Node

   - **INIT STATE:**

     The node sets its transmission power to total network size D, wake up rate $\mu = \mu_{init}$, channel condition $p = \Omega+\delta$ and cluster ID $C_{id} = 1$. When a intermediate node receives a data message, the node goes to the **OP STATE**.

   - **OP STATE:**

     After node receives a data message, node retrieves the traffic rate λ and AIMD command field on the data message. Node updates the wake up rate using AIMD command if command is Additive Increase (AI) then $\mu_i = \mu_i + \Delta$, else $\mu_i = \frac{\mu_i}{\Psi}$.

     When a beacon is received, retrieve information on channel condition p, compare it with previous channel condition. If retrieved information on traffic rate λ or channel condition p is different with previous values, node runs the Iterative Algorithm and determines the optimal number of clusters h, transmission power on distance D/h, wake up rate μ_o, C_{id}. Furthermore, node estimates wake up rate μ_c of the next cluster [15] using the exponential filter. If $\mu_c \leq \mu_o$ sends Additive Increase (AI) command on the data message, else send Multiplicative Decrease (MD)command on the data message. Go back to **INIT STATE**.

2.6.3 Transition Period

All wireless sensor nodes in the network are autonomous, and each node has to decide by itself upon which state to enter. The simple cluster algorithm described in Section 2.6.1 should be able to offer continuous support during a transitory time of a cluster ID. Each node's cluster ID and wake up rate are dependent on channel condition and traffic rate λ. However if nodes do not recognize a change of channel condition or traffic rate before other nodes adapt a new cluster ID, these nodes fail to continue their task. Furthermore, these nodes can generate duplicate packets in the network. Wake up rate and density of nodes are the critical components in our clustering technique. If network has low density of nodes in a region, then network can not adapt to a change of traffic rate or channel condition and gathering of data will not be possible. If

each node has a higher wake up rate, node can easily captures a change using received data or beacon message. However, if a network has a lower packet receiving requirement Ω or low traffic rate λ resulting in low wake up rate, higher density of nodes will fail to adapt the changes in network.

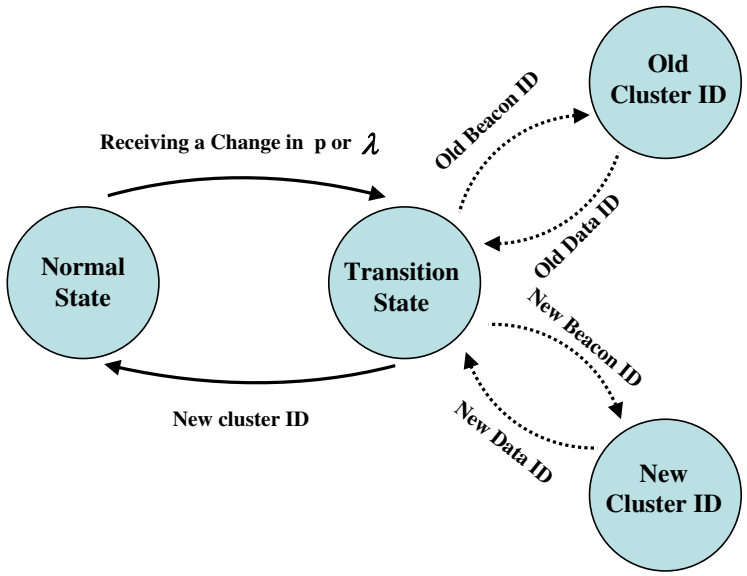

Figure 2.4: State diagram during the transition period

The behavior of a node can be explained considering the state state diagram of Fig. 2.4.

- NORMAL STATE

  When a node receives the same traffic rate λ and channel condition p as the previous ones, the node follows the Randomized Protocol in state machine Fig. 2.1. If node captures a change of traffic rate on a data message or channel condition on a beacon message when the transceiver is in receive mode (i.e., during its wake up period), node runs the Iterative Algorithm and determines the optimal number of clusters h, the transmission power on distance D/h and the optimal wake up rate μ_o, the node uses the simple clustering technique to assign the new cluster ID C_{id}. If the new cluster ID is different with previous cluster ID, node enters the TRANSITION STATE. Otherwise if the new cluster ID is same with previous cluster ID, node keeps the NORMAL STATE. Let us assume that the new cluster ID k matches with previous cluster ID k. Node send a beacon message containing a new channel condition to cluster ID $k + 1$. After node receives a data message, node sends a data message containing a new traffic rate to cluster ID $k - 1$.

- TRANSITION STATE

  Consider the transition state of a node that has to change cluster ID from a old cluster ID i to a new cluster ID j. Node starts a grenade transition timer of a fixed duration called transition period that must be long enough to flood a beacon containing the new channel condition to nodes in previous cluster. Note that the change of hops is independent from the traffic rate in Section 5.4. Node sends a beacon message whose cluster ID is $i + 1$ during the transition period. If the node receives a data message whose cluster ID is i, the node waits a beacon message whose cluster ID is i or j, i.e. when node receives a beacon, node keeps two cluster ID during transition period. If a beacon whose cluster ID i is received, the node goes to the OLD GROUP ID STATE. Otherwise if a beacon whose cluster ID j has been received, the node goes to the NEW GROUP ID STATE. When a transition timer expires, the node no longer needs to consider the old cluster ID i and sets the new cluster ID j and goes to to the NORMAL STATE.

- OLD CLUSTER ID STATE

  After the node sends a data message whose cluster ID is $i - 1$, the node calculates the next sleeping time and goes to sleeping state. When the sleeping timer expires, the node goes to TRANSITION STATE.

- NEW CLUSTER ID STATE

  After the node sends a data message whose cluster ID is $j - 1$, the node calculates the next sleeping time and goes to sleeping state. When the sleeping timer expires, the node goes to TRANSITION STATE.

However, in a low wake up rate of node, this simple cluster technique does not properly address to assign a cluster ID. We modify a wake up rate μ_i of a node to prevent losing node during transition period. The node just takes care of an increase AIMD command on a data message to keep higher wake up rate during a transition period. Furthermore, if a node has lower wake up rate μ_i than μ_{con}, node changes a wake up rate to μ_{con} during transition period. When the node whose state is in transition state receives a different channel condition p or traffic rata λ, node discards a new channel condition p or traffic rate λ, i.e. nodes ignore a change in traffic rate or channel condition during transition period.

2.6.4 Consequence

Distributed Algorithm allows each node to work independently from its neighborhood and to select the next cluster according only to its position and the

positions of source and destination. Furthermore, nodes are not required to maintain a neighbor list and the death of a node is met with an individually determined increase in all its neighbors activity. Consequently, the protocol is extremely robust against topology changes such as node failures and introduction of new nodes.

Chapter 3

Extension of the Randomized Protocol

Even though the Randomized Protocol is a novel protocol to maximize the network lifetime, it still has the some points to be modified for more accurate model and the better performance. The main problems are listed as:

1. Cost function during fixed active time

   Energy consumption during the fixed active time composes the fixed idle listen mode and the cost to send a beacon message. However the energy consumption of beaconing depends on size of clusters in network. Therefore the energy consumption for fixed active time should take account of the size of a cluster.

2. Collisions between beacons and packets

   In the paper [14], Randomized Protocol assumes negligible collision between beaconing and transmitting a data message in network. Therefore, the collision probability will be higher than this expectation.

3. Duplicate packets

   In the Randomized Protocol, all nodes wake up at random time and send a beacons message. Let us assume there are more than one node keeping wake up state in the same cluster, then there is a probability to receive the same packet at the same time. However, Randomized Protocol does not provide the mechanism to avoid the duplicate packets in network.

4. Power control

   After we run simple iterative algorithm in section 2.5, we can derive the optimal cluster size to transmit packet. Therefore power control should consider the radio propagation distance.

We modify the Randomized Protocol to derive more accurate model and improve the performance in overall network. There are two main communication in the Randomized Protocol: receiving and transmitting of beacons and data packet. In Section 3.1, we propose the frequency division access scheme that assigns the different frequency band to beacon and data communication. Frequency division access scheme makes it possible to avoid the collision between beacons and data messages. Section 3.2 provides contention scheme to prevent the duplicate packets. Finally, distance based power control to transmit a data and beacon packet will be described in Section 3.3.

3.1 Frequency Division Access Scheme

The Industrial, Scientific and Medical (ISM) band is widely used among popular wireless network standards such as IEEE 802.15.4 Low-Rate Wireless Personal Area Network (LRWPAN) [22], IEEE 802.11b Wireless Local Area Network (WLAN) [23], IEEE 802.15.3, and BluetoothTM. Because of the mobility and ubiquitous deployment of wireless systems, there are many scenarios where different systems operate in the same place at the same time. Hand-held Personal Digital Assistant (PDA) can use a Bluetooth device to connect to a laptop with 802.11b WLAN. The ISM band is also used by home appliances such as microwave ovens. The microwave oven in the house can be turned on when cordless phone is being used. Furthermore, the recent IEEE 802.15.4 standard for low-rate wireless personal area networks (PANs) is widely considered as one of the technology candidates for wireless sensor networks.

The IEEE 802.15.4 network can operate in any one of three frequency bands, around 868 MHz, around 915 MHz, and the ISM band at 2450 MHz. The maximum data rate depends on the frequency band used; it is 250 kb/s for a network operating in the ISM band. Tmote Sky leverages industry standards like USB and IEEE 802.15.4 to interoperate seamlessly with other devices. Moreover, the CC2420 in the Tmote Sky is a true single-chip 250kbps 2.4 GHz IEEE 802.15.4 compliant RF transceiver designed for low-power and low-voltage wireless applications.

Table 3.1 shows the channel allocations for IEEE 802.11 and IEEE 802.15.4 in the 2.4 GHz ISM band. IEEE 802.11 has 11 channels and each channel has a frequency range of 22 MHz. On the other hand, IEEE 802.15.4 has total 16 channels. Each channel is 5 MHz apart and has a frequency rage of 3 MHz. The frequency values for IEEE 802.15.4 are the central frequency for the channel. The frequency range of each IEEE 802.11 channel is overlapped with the frequency ranges for four different IEEE 802.15.4 channels. For example, IEEE 802.11 channel 6 has the frequency range between 2.401 MHz and 2.423 MHz, which includes the frequency ranges for IEEE 802.15.4 channel 16 through channel 19.

Table 3.1: 2.4 GHz ISM Band, IEEE 802.11 Channels and IEEE 802.15.4 Channels

	IEEE 802.11		IEEE 802.15.4	
	Ch.	Freq.(GHz)	Ch.	Freq. (GHz)
2.4 GHz ISM Band	1	2.401 - 2.423	11	2.405
	2	2.404 - 2.248	12	2.410
	3	2.411 - 2.433	13	2.415
	4	2.416 - 2.438	14	2.420
	5	2.421 - 2.443	15	2.425
	6	2.426 - 2.448	16	2.430
	7	2.431 - 2.453	17	2.435
	8	2.436 - 2.458	18	2.440
	9	2.441 - 2.463	19	2.445
	10	2.446 - 2.468	20	2.450
	11	2.451 - 2.473	21	2.455
			22	2.460
			23	2.465
			24	2.470
			25	2.475
			26	2.480

Because of the overlapped frequency range, IEEE 802.11 channel 6 can cause radio interference to IEEE 802.15.4 channel 16 through channel 19 when it is operating in proximity. The 802.15.4 channel allocation provides a simple provision for coexistence. Channels 25 and 26 use a frequency range outside of the frequency range for 802.11 channels. Therefore, those two channels can be used for a certain environment, in which frequency interference of 802.11 is not expected.

Fig. 3.1 shows the RF output power of the Tmote Sky module from the CC2420 radio. For this test in datasheet [24], the Tmote Sky module is transmitting at 2.405GHz (IEEE 802.15.4 channel 11). The CC2420 programmed output power is set to 0 dBm. The measured output power of the entire modulated spectrum is 2.4 dBm. Since the output power in the frequency 2.41 GHz is less than -45 dBm, frequency division access scheme can prevent the collision between beacons and data messages.

Consequently, we propose the fixed frequency assignment between beaconing and data transmission. Since channels 25 and 26 are not subject to the interference from the IEEE 802.11, we assign the channel 25 for the data transmission and the channel 26 for beaconing.

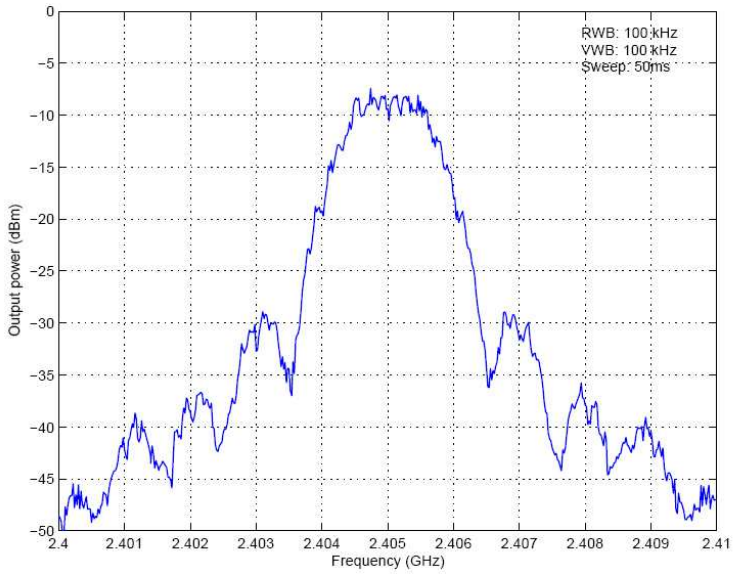

Figure 3.1: Measured RF output power over the modulated spectrum from the Tmote Sky module

3.2 Random Contention Scheme

In the Randomized Protocol, the sender has knowledge of the region to which the packet will be forwarded, but the actual choice of forwarding node is made at random. Due to the random wake up rate, the Randomized Protocol may cause duplicate packets, thus wasting energy in WSNs. We propose simple contention scheme to prevent the duplicate packets. After nodes receive a beacon message from a next cluster, nodes have random delay time. Calling the random contention scheme, the protocol can be summarized as follows:

1. Consider one transmitting cluster and one receiving cluster. Let us assume there are several nodes in transmitting cluster that try to send same data packets to receiving cluster.

2. After these nodes receive the beacon message from receiving cluster, nodes wait a backoff time T_d instead of directly transmitting a data message. The back-off time T_d is a uniformly distributed random variable drawn from 0 to maximum value T_{dmax}.

3. If nodes listen transmission of a data message which has a same sequence number with its own data message within the interval 0 to T_d, the node discards the data packet and goes to CALCULATE STATE. Otherwise, if the random delay timer expires, node transmits a data message to receiving cluster.

With this approach, nodes need to be aware only of the next hop cluster connectivity and do not need a neighbor list of next hop nodes. We believe this is a great benefit because, while neighbor lists of nodes are usually time-varying (nodes may run out of power and other nodes may be added) and hence, their management requires significant overhead, cluster based connectivity is much more stable. In SERAN [17], it uses a similar contention scheme using acknowledgment to reduce the packet duplication effect. But this acknowledgment contention scheme consumes energy to transmit acknowledgment messages and increases the listening time both in the transmitting and receiving cluster. Therefore random contention scheme can be more energy efficient than acknowledgement contention scheme.

However, we still cannot guarantee that duplicate packets are not generated. This may happen if a transmitting node does not hear the data message sent by another node in the same cluster [17]. (i.e., Nodes generate the same random delay T_d.) Furthermore, in case of a collision between same data messages in the same cluster, we lose the data message.

3.3 Power Control on Randomized Protocol

The purpose of this chapter is to investigate on power control for Randomized Protocol. A model of the Received Signal Strength Indicator (RSSI) is proposed and its relation with the packet error rate is studied. A power control algorithm is proposed to find the distance necessary to receive packets with a given error probability.

3.3.1 Radio Propagation Model

The attenuation of the radio power is far from being ideal. The ideal circle used to model the power attenuation can not be used to predict accurately packet losses. In reality, the received power at certain distance is a random variable due to shadow fading and multipath propagation effects, which is also known as fading effects.

We adopt a description of the wireless channel using two components [25]. The first one is known as path loss model, which predicts the mean received power at distance d, denoted by $\overline{Pr(d)}$. It uses a distance d_0 as reference. $\overline{Pr(d)}$ is computed as follows.

$$\frac{Pr(d_0)}{\overline{Pr(d)}} = \left(\frac{d}{d_0}\right)^{\beta} \tag{3.1}$$

Where the $PL(d_0)$ is the path loss at a reference distance d_0, and β is called the path loss exponent, and is usually empirically determined by field mea-

surement. Larger values correspond to more obstructions and hence faster decrease in average received power as distance becomes larger. Path loss exponent is normally in the range of 2 to 4 (where 2 is for propagation in free space, 4 is for relatively lossy environments and for the case of full specular reflection from the earth surface). In some environments, such as buildings, stadiums and other indoor environments, the path loss exponent can reach values in the range of 4 to 6. On the other hand, in tunnels a waveguide type of propagation may occur with the path loss exponent dropping below 2.

The second part of the shadowing model reflects the variation of the received power at certain distance. It is a log-normal random variable, that is, it has a Gaussian distribution if measured in dB.

The overall attenuation model is represented by

$$\left[\frac{Pr(d)}{Pr(d_0)}\right]_{dB} = -10log\Big(\frac{d}{d_0}\Big) + X_\sigma \tag{3.2}$$

where X_σ is a Gaussian random variable with zero mean and standard deviation σ, which is called the shadowing deviation, and is obtained by measurement.

The shadowing model extends the ideal circle model to a statistic model: nodes can only probabilistically communicate when near the edge of the communication range.

3.3.2 Distance based Power Control

Consider a node transmitting packets with a radio power level P_t. The distance at which packets are successfully received depends on the modulation, encoding, output power, frame size, noise floor, and channel parameters. By exploiting the attenuation characteristics of a transmitted signal, we can determine the distance at which a packet is successfully received by controlling the transmit power. Specifically, by calling Eq. (3.1) the attenuation, in dB, of a wireless signal at the generic distance d from the transmitter can be expressed as follows:

$$PL(d)_{\,dB} = PL(d_0)_{\,dB} + 10\,\beta\,log_{10}(\frac{d}{d_0}) + X_\sigma\ , \tag{3.3}$$

The signal attenuation, therefore, follows an exponential decay with respect to distance, and a Gaussian attenuation having zero average and standard deviation σ. Since the term $PL(d)_{\,dB}$ has a Gaussian distribution, the received power P_r, in linear unit, follows a log normal random distribution.

It is easy to see that the Signal to Interference plus Noise Ratio in dB can be written as follows [26]:

$$\gamma(d)_{\,dB} = P_r(d)_{\,dB} - P_{ni\,dB} \tag{3.4}$$

where $P_r(d)$ is the received power:

$$P_r(d)_{\,dB} = P_t(d)_{\,dB} - PL(d)_{\,dB}\,, \tag{3.5}$$

and P_{ni} denotes the noise floor and interference.

The Chipcon CC2420 uses the O-QPSK (offset quadrature phase shift keying) modulation with DSSS. The bit error probability for O-QPSK modulation schemes with coherent demodulation in a slow Rayleigh fading environment, which exhibits non-selective behavior both in frequency and in time [27], can be expressed by:

$$P_b(d) = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma}(d)}{1+\overline{\gamma}(d)}}\right) \tag{3.6}$$

where $\overline{\gamma}(d)$ is the average SINR at a the distance d. To compute the bit error probability Eq. (3.6), we need to characterize the average SINR. Since the SINR shows a log normal distribution, it can be proved that its average value is given as follows:

$$\overline{\gamma}(d) = e^{\mu_\gamma + \sigma_\gamma^2/2} \tag{3.7}$$

where μ_γ and σ_γ are, respectively, the average and standard deviation of the SINR in neper unit. By recalling the relation of the neper units with the dB units, and Eq. (3.7), the following relation holds true:

$$\mu_\gamma = \frac{log_e(10)}{10}\left[P_{t\,dB} - PL(d_0)_{\,dB} - 10\,\alpha\, log_{10}(\frac{d}{d_0}) - P_{ni\,dB}\right] \tag{3.8}$$

and

$$\sigma_\gamma = \frac{log_e(10)}{10}\sigma \tag{3.9}$$

Given the packet size l containing the Synchronization Header and PHY Header, MAC Header, MAC Footer in [24], the probability of successful packet reception at a distance d, which we denote with $P_{succ}(d)$, is

$$P_{succ}(d) = \Big(1 - P_b(d)\Big)^l \tag{3.10}$$

By imposing a constraint on the probability of successful packet reception at a distance d, P_{con}, we can translate the constraint on the average SINR by Eq. (3.6) and (3.10), thus obtaining a $\overline{\gamma}_{con}$. From these we can derive the transmit radio power necessary to successfully receive packets at a distance d with probability P_{con}. After simple algebra, we have that

$$P_t(d)\;dB = \overline{\gamma}_{con\;dB} + PL(d_0)_{\,dB} + 10\,\beta\, log_{10}(\frac{d}{d_0}) + P_{ni\,dB} - \frac{log_e(10)}{20}\sigma^2 \tag{3.11}$$

Where the constraint on the average SINR is $\overline{\gamma}_{con} \geq \frac{\left(2P_{con}^{1/l}-1\right)^2}{1-\left(2P_{con}^{1/l}-1\right)^2}$.

3.3.3 Empirical Analysis of the CC2420 Transceiver

Path Loss Prediction Model

From the Eq. (3.11), we can compute optimal transmission power where $PL(d_0)$, α, σ are found experimentally. In our experiments we use Tmote Sky [20] that are based on the CC2420 radio chip [24], which is based on IEEE 802.15.4 [22].

CC2420 RF Transceiver

The Chipcon CC2420 [24] on the Tmote Sky IEEE 802.15.4 radio transceiver operates in the $2.4GHz$ ISM band, and includes a digital direct sequence spread spectrum (DSSS) modem. In the $2.4GHz$ band, it has 16 channels (numbered 11 through 26) with each channel occupying a $3MHz$ bandwidth with a center frequency separation of $5MHz$ for adjacent channels. CC2420 uses an encoding scheme that encodes 32 chips for a symbol of 4 bits. This encoded data is then OQPSK (offset quadrature phase shift keying) modulated. It provides a spreading gain of $9dB$ and an effective data rate of $250Kbps$, a much higher rate than older radios. It has been specifically designed for low power wireless applications and supports 8 radio power levels: $0dBm$, $-1dBm$, $-3dBm$, $-5dBm$, $-7dBm$, $-10dBm$, $-15dBm$ and $-25dBm$ at which its power consumption varies from $29mW$ to $52mW$ in Table 3.2.

Table 3.2: Output power settings and Typical current consumption in 2.45 GHz

PA_LEVEL	TXCTRL register	Output Power	Current Consumption
31	0xA0FF	0 dBm	17.4 mA
27	0xA0FB	-1 dBm	16.5 mA
23	0xA0F7	-3 dBm	15.2 mA
19	0xA0F3	-5 dBm	13.9 mA
15	0xA0EF	-7 dBm	12.5 mA
11	0xA0EB	-10 dBm	11.2 mA
7	0xA0E7	-15 dBm	9.9 mA
3	0xA0E3	-25 dBm	8.5 mA

CC2420 provides the useful measurements can be referred as a Received Signal Strength Indicator (RSSI) [28]. It is an estimate of the received signal power within the bandwidth of an IEEE 802.15.4 channel. The receiver energy

detection's result shall be reported over 8 symbol periods and stored in the RSSI.VAL register.

Chipcon specifies the following formula to compute the received signal power $P_{r\ dBm}$ from CC2420 datasheet [24]:

$$P_{r\ dBm} = \text{RSSI.VAL} + \text{RSSI.OFFSET} \tag{3.12}$$

where RSSI.OFFSET can be found empirically during development from the front end gain (approximately $-45\ dB$). The data sheet for the CC2420 states that the accuracy of the RSSI reading is $\pm 6\ dB$.

Experimental result

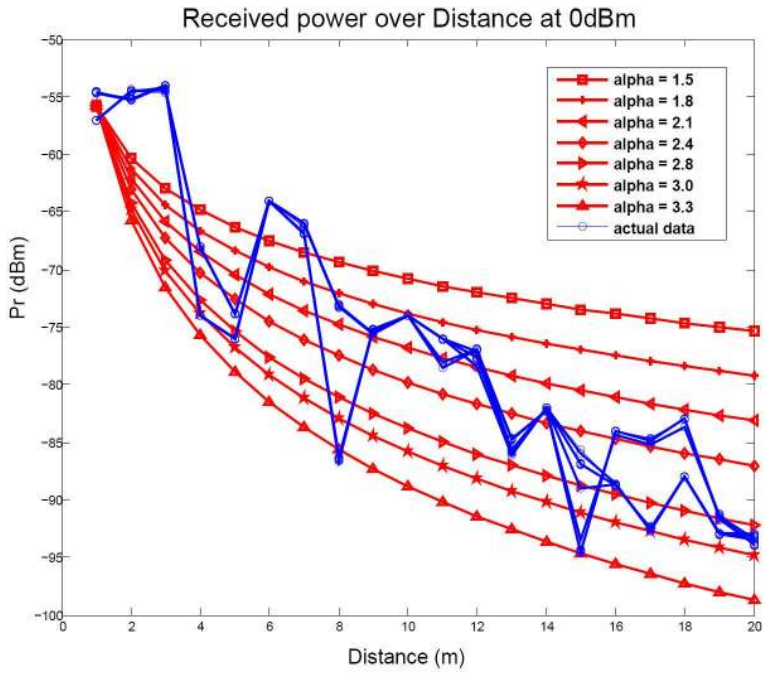

Figure 3.2: Received power over distance

We carried out an evaluation of the on a Tmote Sky provided by the Moteiv [20]. In a practical world, the system is hindered by a number of parameters and this section is to investigate the effect of these parameters on the ideal system. Throughout testing it was assumed the response of the RSSI measurements given by the CC2420 were linear in [24]. It was also assumed environmental conditions were constant over the measuring period.

In the long open corridor, since we experimented the Randomized protocol on the floor, a receiver and transmitter were placed on the floor. A transmitter was placed at 1 meter intervals along a straight line away from the receiver. Throughout the experiment the receiver was not moved. At each location the transmitter was placed in the same orientation to avoid non-linearities in transmission pattern. The transmitter sent the packet at the rate of 2 packets every 1 second in channel 11 and at a particular transmission power level $0\ dBm$. When the receiver received a packet, it read RSSI.VAL register. This process was repeated for 0 to 20 meter intervals. After the receiver corrected 1000 packets at each distance, we estimated the model parameters using sample averages.

The graph 3.2 shows the average RSSI each 100 packets over the $20m$ range for a fixed transmission power $0\ dBm$. From it we can clearly observe a Log normal relationship as expected. It is also possible to fit the curve to a path loss exponent 2.8.

To asses the suitability of using the RSSI readings in a power control a plot of the standard deviation of the readings is shown in Fig. 3.3. We can see that there is no particular distance that gives an outstandingly bad performance over this range. The largest recorded standard deviation σ in readings was measured to be 9.

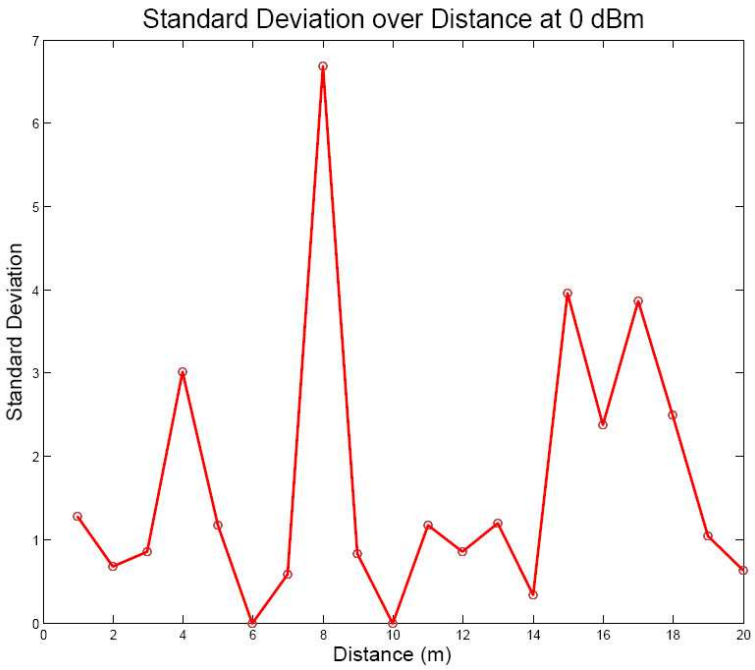

Figure 3.3: Standard deviation over Distance

Chapter 4

Hardware and Software Platform

This chapter describes the whole setup of experiments in terms of time synchronization, hardware, software and how the Randomized Protocol experiments where performed.

4.1 Time Synchronization

Time synchronization is a critical aspect for any distributed system. Distributed, wireless sensor networks make extensive use of synchronized time, but often have unique requirements in the scope, lifetime, and precision of the synchronization achieved, as well as the time and energy required to achieve it [5].

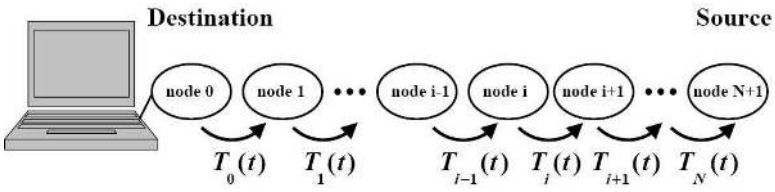

Figure 4.1: Sensor field

Fig. 4.1 shows our scenario for experiment that have destination node whose number is 0, source node whose number is $N+1$ and N number of intermediate nodes. Time synchronization for Randomized Protocol is simple scheme to compute the end-to-end delay in network. The main idea of the time synchronization is flooding the global time from the destination node to the source node. The process of the time synchronization follows these steps:

- All nodes maintain WAKE UP state to synchronize the time during the initial state in network.

- In the Fig. 4.1, destination node sends synchronization packet (SYNC-packet) containing the first global time $T_0(t)$ which is the local time of the destination node to node 1 closest to destination node using unicast.

- Consider the node number i. After the node i receives time SYNC-packet from node $i-1$, node initializes its own local time to received global time $T_{i-1}(t)$. When the node i finishes the synchronization process, it generates new global time $T_i(t)$ and unicasts the synchronization packet to next node i+1. New global time $T_i(t)$ compensates the synchronization processing time $\Delta_{i,i-1}$ in the node i, $T_i(t) = T_{i-1}(t) + \Delta_{i-1,i}$.

- After the source node $N+1$ receives the SYNC-packet from last intermediate node N and initializes the time, time synchronization process in network is completed.

Consequently, the source node $N+1$ receives SYNC-packet containing the global time $T_N(t) = T_0(t)+\Delta_{0,1}+\Delta_{1,2}+\ldots+\Delta_{N-1,N}$. If we let the propagation time $\epsilon_{i,i+1}$ from the node i to the node i+1, then the error-free last SYNC-packet for source node should be defined as:

$$T_{N,true}(t) = T_0(t) + \underbrace{\epsilon_{0,1} + \epsilon_{1,2} + \cdots + \epsilon_{N-1,N}}_{total\ propagation\ delay} + \underbrace{\Delta_{0,1} + \Delta_{1,2} + \cdots + \Delta_{N-1,N}}_{total\ processing\ delay}. \tag{4.1}$$

Therefore, we can estimate the synchronization error $T_{error}(t)$:

$$\begin{aligned} T_{error}(t) &= T_{N,true}(t) - T_N(t) \\ &= \epsilon_{0,1} + \epsilon_{1,2} + \cdots + \epsilon_{N-1,N} \end{aligned} \tag{4.2}$$

The reasons of delay can be split into two parts, propagation delay and processing time on each nods. From Eq. (4.2), this time synchronization compensates the total processing time in the network. Since small size WSNs has a negligible propagation delay per hop, simple time synchronization works well.

Finally, Time synchronization in Randomized Protocol can be summarized as follows: When node receives synchronization packet stamped with global time, node synchronizes with this global time and generates new global time to compensate processing time. Node unicasts new global time to next node and this process repeats to source node. When the source node receives the synchronization packet, time synchronization in the network is completed.

4.2 Hardware Platform : Moteiv Tmote sky

Tmote Sky is a node platform for low power, high data-rate sensor network applications designed with the dual goal of fault tolerance and development ease [20] [29]. It is the successor of the popular TelosA and TelosB research platforms. Designed at the University of California, Berkeley, be TinyOS developers, the Tmote sky platform offers vertical integration between the hardware and the TinyOS operating system. Tmote Sky is a FCC Certified WSN platforms available on the market, making it particularly suitable for Original Equipment Manufacturer (OEM) integration. It has integrated sensors, radio, antenna, microcontroller and programming capabilities.

The low power operations of the Tmote Sky module is due to the low power TI MSP430 F1611 microcontroller. This 16-bit RISC processor features low active and sleep current consumption. In order to minimize power consumption, the processor in sleep mode during majority of the time, wakes up as fast as possible to process, then returns to sleep mode again. Tmote Sky provides an easy-to-use USB protocol from FTDI to communicate with the host computer for programming, debugging and data collection. It features the Chipcon CC2420 radio [24], which is an IEEE 802.15.4 [22] compliant radio providing reliable wireless communication. The radio provides fast data rate and robust signal. It is controlled by the microcontroller through the SPI port and can be shut off for low power duty cycled operation. The internal Inverted-F microstrip antenna is a pseudo omni directional antenna that may attain 50 meter range indoors and up to 125 meter range outdoors.

1. MicroController Unit (MCU)

   The Tmote Sky is equipped with a MSP430F1611 microcontroller from the TI MSP430 family of low- power 16-bit microcontrollers. The MSP430 family comes with a AD converters, ports, universal asynchronous receiver/transmitter (uart), Serial Peripheral Interface Bus (SPI), Inter Integrated Circuit (I2C) buses. On the Tmote much of this can easily be connected to external devices. The on-board microcontroller offers 10k byte RAM and 48k byte flash programmable ROM. The board also comes with 1M byte of external flash memory connected to the SPI bus.

2. Radio Transceiver

   The Tmote Sky is equipped with CC2420, an IEEE 802.15.4 compliant radio transceiver from Chipcon/ Texas Instruments. The transceiver is connected to an internal inverted-F antenna giving it about 50 m of useful communication range when used indoors, and upwards of 125 m outdoors. The maximum RF output power of the platform is about 0 dBm.

3. External Storage

The Tmote Sky has a 1 MB serial flash, M25P80, that can be read and written by the MSP430. Tmote Sky provides hardware based write protection to parts of the flash that is disabled only when the module is powered via the USB interface. This enables storing a protected factory image which can not be erased/modified during normal node operations.

4. Sensors

   In addition to the internal temperature sensor of the MSP430 microcontroller, the Tmote Sky board has predefined positions for mounting a humidity/temperature sensor from Sensirion AG (models SHT11 and SHT15 are supported), as well as for light sensors like the Hamamatsu Corporation S1087 for sensing photosensitive solar radiation or the S1087-01 for total spectrum measurements.

5. Energy Storage

   Battery The Tmote Sky is powered by two AA size (1.5 V) batteries affixed in a standard battery pack. The operating range of the platform when on battery power is from 2.1 to 3.6 V, with 2.7 V needed for internal and external flash reprogramming.

   External Power-Supply Interfaces The Tmote Sky can also be powered via the on-board Universal Serial Bus (USB) interface when plugged into the USB port of a host computer for programming or communication. The operating voltage when attached to the USB is 3 V.

6. External Interfaces

   Data Buses The Tmote Sky uses a FT232BM usb-to-serial chip from FTDI as a primary communication channel with the host controller. The USB interface is connected to the USART1 module on the microcontroller. On the PC side, the USB connection appears as a regular serial port. In addition to the USB serial channel, the expansion connector on the Tmote Sky also exports the raw UART0 receive and transmit lines and one I2C and/or SPI bus.

   Debug/Programming The same USB interface is also used to (re)program the microcontroller flash. Tmote Sky exports the microcontroller Joint Test Action Group (JTAG) pins for in-circuit debugging and reprogramming via an additional interface. The microcontroller flash can also reprogram itself from software (with some OS support).

   ADC Inputs The 10-pin extension connector provides access to four ADC channels. Additional two channels are exported via the 6-pin "exclusive" interface.

Digital I/O and Interrupts The extension connector provides four pins that can be used as general I/O (when not used as ADC/I2C functional pins). Two additional general IO pins are available on the "exclusive" interface. This interface also exports the two DAC channels of the microcontroller as well as the interrupt lines for the user interface elements, the reset and the user buttons.

7. Packaging

   Dimensions Tmote Sky has the following nominal dimensions: width : $3.2cm$, length : $6.55cm$ and height : $0.66cm$.

8. Availability

   Developer/Manufacturer The Tmote Sky boards are produced by MoteIV Corporation, San Franciso, CA.The units can be obtained directly from the manufacturer via a convenient web ordering service. Purchased individually, a single Tmote Sky unit (without the on-board sensors) costs about $130. The humidity, temperature and light sensors cost additional $50. A 10-unit Tmote Sky Developer Kit can be obtained for $790, driving the single unit price down to $79.

9. Support

   The main support for the Tmote Sky platform is provided via the manufacturer web site and via e-mail. Software related issues are also handled via the TinyOS mailing lists since many of the Tmote Sky users are also using TinyOS.

   Software Tmote Sky is fully compatible with TinyOS 1.x and TinyOS 2.0 and MoteIV originally distributed a cygwin based software installer providing full TinyOS development environment for Windows. Recently, they have developed a hardened TinyOS distribution called Boomerang, specifically optimized for the MoteIV product family.

   Documentation When delivered to the customers, the Tmote Sky units are accompanied with a "Quick Start Guide" and a detailed "Data Sheet" documents. New versions of the documentation can be downloaded from the MoteIV web site [30].

4.3 Operating System : TinyOS

The one of the key elements of the research platform in WSNs is the operating system [29] [31] [32]. Unfortunately, most software design patterns focus on the problems faced by large, object-oriented programs: the challenges sensor

network programs face are quite different [33]. Therefore operating system in WSNs should fulfill these requirement:

- Robustness: once deployed, a sensor network must run unattended for months or years;
- Low resource usage: sensor network nodes include very little RAM, and run off batteries;
- Diverse service implementations: applications should be able to choose between multiple implementations;
- Hardware evolution: mote hardware is in constant evolution; applications and most system services must be portable across hardware generations;
- Adaptability to application requirements: applications have very different requirements in terms of lifetime, communication, sensing, etc.

Because of the extremely limited resources of the hardware platforms, it is difficult to virtualize system operation to create the kinds of system abstractions that are available in more resource rich systems. The concurrency model and abstractions provided by operating system therefore significantly impact the design and development process.

TinyOS was used as the development environment in this thesis, which is one of the first software environments specifically designed to meet the requirements of resource-constrained, event-driven and networked embedded systems [31], [32]. Originally developed by the University of California, Berkeley and Intel at the beginning of the decade, it has since become the most popular operating system for Wireless Sensor Networks. The TinyOS 1.x family is the latest stable branch of the operating system and is used in this section to describe the basic design principles. The TinyOS development environment directly supports a variety of device programmers and permits programming each device with a unique address attribute without having to compile the source code each time. The TinyOS system, libraries and applications are written in nesC, a Version of C that was designed for programming embedded systems. The characteristics of ThinyOS 1.x are listed as:

1. Component-based modularization

   Many features of TinyOS stem from the design goals of its implementation language called nesC. One of the main characteristics of its programming model is the use of component modularization. In this model, the functionality of the traditional monolithic abstraction layers is broken-up in smaller, self-contained building blocks that interact with each other

via clearly defined interfaces. This hiding of the implementation behind well-defined interfaces preserves the modularity of the solution and promotes reuse. At the same time, the component model supports richer interactions between the building blocks.

2. Concurrency model

   Equally important to the type of code organization is the nature of the supported interactions between the components. The main concern here is how to best facilitate the asynchronous and event-driven type of exchange that occurs not only in the communication context, but also in the user space, as the applications in WSNs are tightly coupled with the environment and usually perform processing as a reaction to some sensed event. Dealing with the dataflow-centric nature of the applications using limited hardware resources requires making smart trade-offs in the operating system design. TinyOS attacks the problem by offering two levels of concurrency: tasks and events. Tasks are a mechanism for deferred computation that should be used whenever the timing requirements for the computation are not strict. This includes almost all application processing apart from the low-level, hardware related operations. Components can post a task, after which the execution immediately returns to the poster. The actual execution is delayed until the task scheduler executes the task later. Due to the high overhead of context-switching, the tasks run to completion and do not preempt each other. Consequently, they have to be kept short in order to guarantee low task execution latency and high system responsiveness. Events also run to completion, but can preempt the execution of tasks and/or other events. A large part of the processing in TinyOS is triggered by receiving events representing hardware interrupts. Events are also used to signalize the completion of a split-phase operation.

3. Split-phase operations

   Due to the non-preemptive nature, TinyOS does not support blocking operations. This means that all long-latency operations have to be realized in a split-phase fashion, by separating the operation-request and the signaling of the completion. The client component requests the execution of an operation using command calls which execute a command handler in the server component. The server component signals the completion of the operation by calling a an event handler in the client component. In this way, each component involved in the interaction is responsible for implementing part of the split-phase operation. The decision not to hide the split-phase nature of the long-latency operations has the benefit of forcing the programmer to be aware of their effect on the responsiveness

of the system. At the same time, this feature turns out to be one of the biggest stumbling blocks for the novice TinyOS users that are used to the more traditional, linear execution model.

4. Static program analysis

   The simple concurrency model of TinyOS allows high event-handling throughput while keeping the overhead significantly lower than in the traditional thread-based approaches. At the same time, the model is still vulnerable to the typical programing errors occurring in concurrent systems including deadlocks and dataraces. Minimizing the introduction of such errors is particularly important in the domain of embedded systems where there is no human-operator in the control-loop that can take mitigating actions. Thus, it is of paramount importance that such errors are detected and corrected prior to execution time. NesC imposes several restrictions on the programmer that reduce the chance of introducing complex bugs in TinyOS code. It is a static language that does not support dynamic memory allocation or function pointers (even if the use of malloc or free in code do not generate compilation errors). Consequently, the call-graph of nesC programs is fully known at compile-time, allowing nesC to perform whole-program analysis for safety and performance optimization. This analysis can detect and report almost all potential dataraces (any update of shared state from asynchronous code). Using this information, the programmer can remove the error by moving the critical sections into tasks or by preventing concurrent-access using atomic blocks.

5. Networking services

   The active messages are the main communication abstraction in TinyOS. They consist of a small identifier that is attached to each message, specifying the action that needs to be taken on the receive side when a given message has been received. In practice the active messages are used similar to the port concept in the TCP/IP stack. Each application has a subset of AM indexes reserved and its receive handlers are triggered whenever the networking stack receives such a message. The lower parts of the networking stack are generally encapsulated in an abstraction called GenericComm that provides single hop unicast and broadcast service. The GenericComm exposes the SendMsg and ReceiveMsg interfaces for sending/receiving fixed size message buffers – TOS Msg. The abstraction does not support send buffering. On that a new message can be received while the user components are processing the current message.

6. Hardware abstraction

Abstracting the capabilities of the hardware is a basic OS responsibility. One of the goals of TinyOS was to facilitate easy traversing of the SW/HW boundary by allowing some SW components to be replaced with real HW modules (that export the same interfaces) and vice versa.

4.4 Experimental Setup

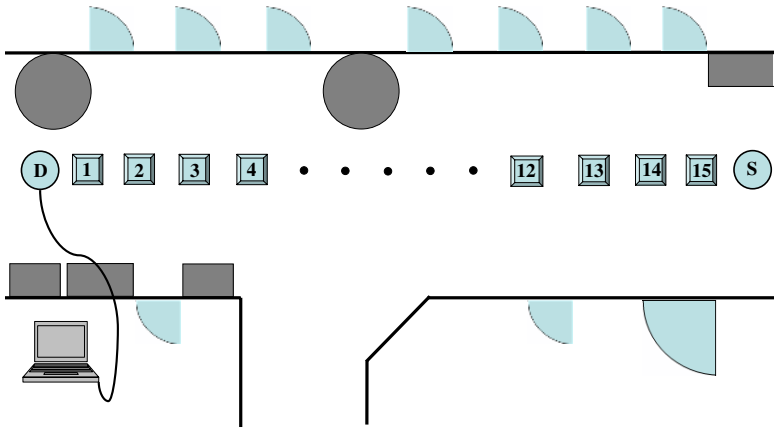

Figure 4.2: Physical disposition of the sensors in the corridor

Our experiment was based on the Extended Randomized Protocol using Tmote Sky, as described in Chapter 2 and 3. Fig. 4.2 shows the office corridor layout in which we built the sensor network. Sensor node "D" is the destination node connected to a PC, node's numbers from 1 to 15 were intermediate nodes, and node "S" was the source. The long distance corridor ($20m$) is surrounded by static objects with minimal time-varying changes in the wireless channel due to multi-path fading effects. We used 17 Tmote Sky sensor nodes, where 15 intermediate nodes were placed at regular intervals ($1.25m$) along a straight line away from the source to destination node. The source generated data message with different traffic rate λ at the distance $20m$ from the destination node. At each location the sensor nodes were placed in the same orientation to avoid non-linearities in transmission pattern.

Chapter 5

Results from the Experiment

Several experiments of the Extended Randomized Protocol implementation were conducted in order to evaluate its performance. This chapter presents the experimental results and observations about the strengths and limitations of the Extended Randomized Protocol. All experiments described in this chapter use the network setup described in Section 4.4. It should be noted that the conclusions drawn in this chapter are based on more experimental data than is shown here. In Section 5.1, we analyze the experimental results in terms of E2E delay guarantee, and error rate guarantee in Section 5.2. Section 5.3 presents performance of random contention scheme to prevent duplicate packets. The last Section 5.4 investigates the lifetime of sensor network, respectively.

5.1 End-to-End delay

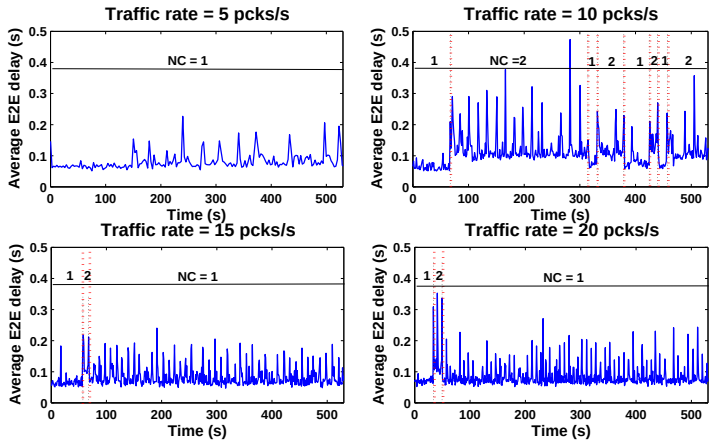

Figure 5.1: Average E2E packet delay vs. Traffic rate

Fig. 5.1 shows the average E2E delay over a experimental period with different traffic rate λ. In the figure, NC is the number of clusters in the network. We set a constraint of error rate $\Omega = 0.6$ and a delay constraint of $\tau = 5s$. The plot shows that the average E2E delay is independent from traffic rate λ because of higher wake up rate in the case of higher traffic rate, and vice versa. Note that the error rate is the dominant constraint to compute optimal wake up rate μ_o with respect to E2E delay constraint. Eq 2.15 shows that the optimal wake up rate of Extend Randomized Protocol is proportional to traffic rate λ. Variations in the delay mainly comes from waiting time of beacon messages. In addition, in Fig. 5.1, we can observe the average E2E delay as function of the number of clusters. The Eq. 2.5 shows the waiting time to receive a beacon message increases as the number of clusters.

Table 5.1: End-to-End delay results

Ω	Traffic rate	Mean	Standard Deviation	Maximum	Minimum
0.6	5 pcks/s	0.0823 s	0.0566 s	0.6062 s	0.0313 s
	10 pcks/s	0.1127 s	0.0943 s	0.8516 s	0.0148 s
	15 pcks/s	0.0825 s	0.0678 s	1.1507 s	0.0276 s
	20 pcks/s	0.0861 s	0.0762 s	2.0722 s	0.0220 s
0.7	5 pcks/s	0.0964 s	0.1025 s	2.0633 s	0.0295 s
	10 pcks/s	0.0820 s	0.0686 s	0.7103 s	0.0161 s
	15 pcks/s	0.0763 s	0.0600 s	1.0267 s	0.0229 s
	20 pcks/s	0.0814 s	0.0671 s	2.0669 s	0.0262 s
0.8	5 pcks/s	0.0765 s	0.0557 s	0.5128 s	0.0286 s
	10 pcks/s	0.0758 s	0.0600 s	0.5302 s	0.0269 s
	15 pcks/s	0.0744 s	0.0640 s	1.4356 s	0.0250 s
	20 pcks/s	0.0843 s	0.0632 s	0.7204 s	0.0172 s
0.9	5 pcks/s	0.0731 s	0.0566 s	0.9722 s	0.0152 s
	10 pcks/s	0.0674 s	0.0520 s	2.0542 s	0.0021 s
	15 pcks/s	0.0769 s	0.0608 s	0.8685 s	0.0197 s
	20 pcks/s	0.0752 s	0.0510 s	2.0619 s	0.0164 s

In Table 5.1, we report the mean, the standard deviation, the minimum and the maximum of the E2E delay as function of the error rate constraint Ω and the traffic rate λ of the source. The Extended Randomized Protocol provides low E2E delay than delay constraint $\tau = 5s$, independent by traffic rate λ and error rate Ω. Therefore we can conclude that the Extended Randomized Protocol satisfies the protocol specifications in terms of latency.

5.2 Error rate

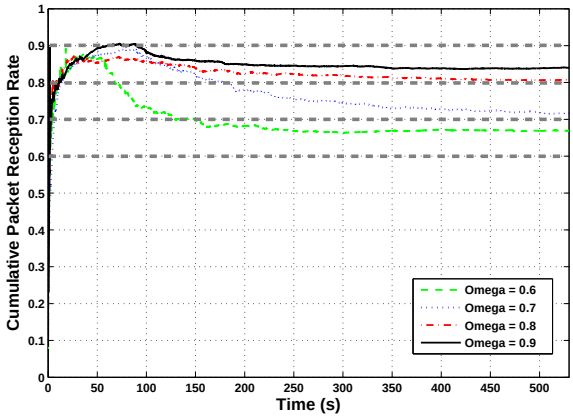

Figure 5.2: Cumulative Packet Reception Rate vs. Error rate constraint

One of main problem in WSN is the high message loss rate, which prevents the system from working properly from time to time. The second set of experiments investigate the message loss properties of Extended Randomized Protocol. In Fig. 5.2 we report the E2E Packet Reception Rate (PRR) for the packet delivery as function of time. We set the traffic rate $\lambda = 10pcks/s$, delay constraint $\tau = 5s$, and four values of the Ω constraints. Figure indicates that Extended Randomized Protocol guarantees the successful packet reception rate. In addition, Fig. 5.2 shows that after destination node starts the first estimation of channel condition, the network converges to a stable error rate.

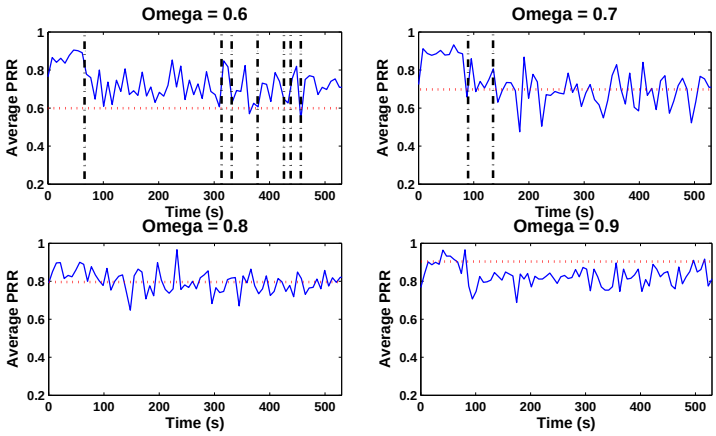

Figure 5.3: Average Packet Reception Rate vs. Error rate constraint

Fig. 5.3 shows average PRR where traffic rate λ is $10pcks/s$ and delay con-

straint $\tau = 5s$. When network has good channel condition, Extended Randomized Protocol increases the number of clusters to decrease energy consumption, and vice versa. Note that transmission power consumption is proportional to cluster size. However, increasing the number of clusters in network gives the outstandingly bad PRR. Therefore, there are less cluster changes in case of higher error rate constraint Ω.

Table 5.2: Cumulative PRR results

Ω	Traffic rate($pcks/s$)	Cumulative PRR
0.6	5	0.68
	10	0.73
	15	0.67
	20	0.62
0.7	5	0.77
	10	0.72
	15	0.70
	20	0.67
0.8	5	0.82
	10	0.81
	15	0.77
	20	0.68
0.9	5	0.86
	10	0.84
	15	0.81
	20	0.68

The results of cumulative PRR are summarized in Table 5.2 as function of the error rate constraint Ω and the traffic rate λ. Looking at Table 5.2 and Fig. 5.2, it is evident that it is hard to achieve a good PRR where $\Omega = 0.9$ and high traffic rate.

There are several reasons:

- Last hop

  We notice that most of messages where lost in the last hop. This is possible due to the fact that $\Omega = 0.9$ or high traffic rate causes high packet collisions in the last hop, where the destination node is only one.

- Required Wake-Up Rate

  Fig. 5.4 shows required optimal cumulative wake up rate μ_o over the channel condition p and maximum allowed wake up rate where the traffic rate λ is $10pcks/s$ and required PRR is 0.9. If channel condition p is smaller than required PRR Ω, then the system needs the infinite wake up

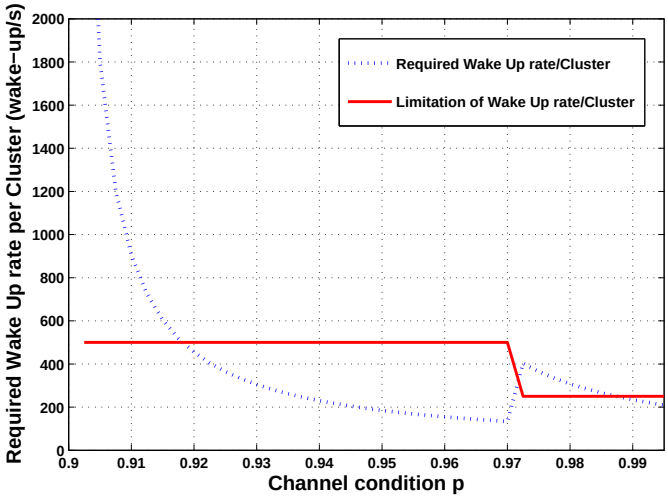

Figure 5.4: Required optimal wake up rate per cluster

rate (see Eq. 2.15). However, the maximum wake up rate in a cluster is $\frac{N}{T_{ac}\,(h-1)}$ where $h-1$ is the number of clusters and N is total number of nodes. Hence, from Fig. 5.4, it can be concluded that the maximum wake up rates in a cluster is not enough to satisfy the required one.

Furthermore, another reason of bad PRR is due to the fact that if the WSN experiences bad channel conditions, then the transmission period during cluster changes will work badly. Therefore, bad channel condition will cause some nodes to be out of the system, thus increasing the packet losses.

5.3 Duplicate packets

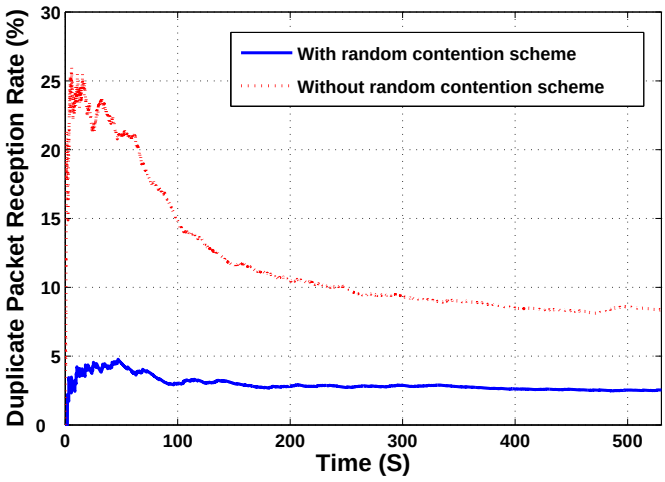

Figure 5.5: Duplicate Packet Reception Rate

In order to take into account the duplicate packets, we use the random contention scheme presented in Section 3.2. Fig. 5.5 shows duplicate packet reception rate with and without random contention scheme, where traffic rate is $15pcks/s$ and error rate constraint $\Omega = 0.7$. Although the random contention scheme can not completely avoid the problem of duplicate packet, this simple mechanism is useful to decrease this phenomenon.

There are two main reasons for which random contention scheme is not able to avoid duplicate packets:

- Fail to recognize

  This may happen if a transmitting node does not hear the data message from another node's transmission which is sending the same data message.

- Too high wake up rate

  If network has higher wake up rate than optimal wake up rate μ_o, extra nodes can receive the same data message.

5.4 Estimation of the Average Energy Consumption

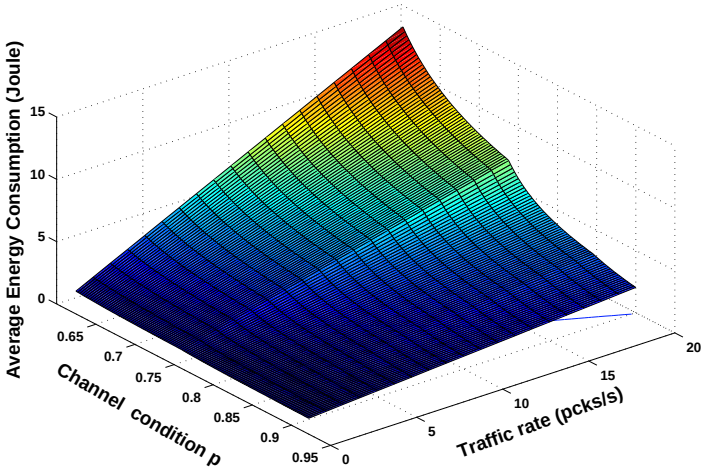

Figure 5.6: Average energy consumption vs. Traffic rate

A primary requirement of deploying wireless sensor networks is a system lifetime. The analysis presented in this section aims at giving an indication of the energy consumption of sensor nodes in the network. This indication enables us to estimate the period after which the batteries of the nodes will have to be replaced and thus the expected system's lifetime.

In Fig. 5.6, the average energy consumption of the whole network as function of traffic rate λ and channel condition p. The error rate constraint is set to $\Omega = 0.6$ and traffic rate is from $0pcks/s$ to $20pcks/s$. Note that estimation of average energy consumption is based on Eq. 2.16. Therefore this estimation provides the upper bound of average energy consumption of the network. If channel condition p is smaller than Ω, then the network consumes the maximum energy to achieve the error rate constrain in Eq. 2.15. Estimated average energy consumption is combined with the two convex surfaces related to channel conditions. A turning point between the two convex surfaces means that the optimal number of clusters changes in the network. Note that when the most stringent constraint is the error rate, the optimum number of hops does not depend on traffic rate λ.

Fig. 5.7 shows average energy consumption over the channel condition p and error rate constraint Ω, where we fix the traffic rate $\lambda = 20pcks/s$. Note that maximum value of channel condition (X axis) is 1. The turning points means that the optimal number of clusters changes in network, as for Fig. 5.7. We find that changes of cluster comes from the channel condition p and required PRR.

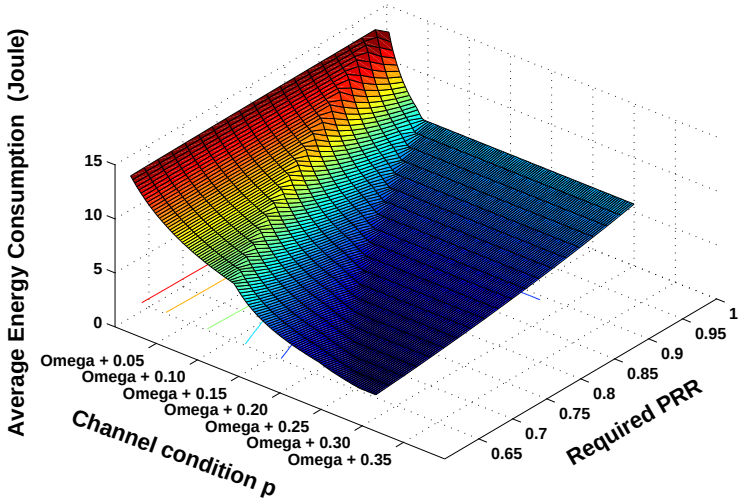

Figure 5.7: Average energy consumption vs. Required PRR

In most of sensor network applications, if a sufficient number of nodes run out of energy, it may impair the ability of the sensor network to function. Moreover, if a node has depleted its energy, there must be a hole in the system and thus incur network partitioning. So load balancing is a critical issue in the design of sensor network. Extended Randomized Protocol provides a load balancing delivery scheme because of random wake up time. Therefore we can assume that energy consumption in network is uniformly distributed between nodes. Consequently, the lifetime of network is proportional to total number of deployed nodes in the network.

Chapter 6

Conclusions

6.1 Conclusions of the work

In this thesis, research on cross layer protocols for WSN in wireless automation is described. We have presented the Extended Randomized Protocol solution for wireless networks that were successfully applied in an indoor environment.

Although the Randomized Protocol is provided in paper [14], performs well in simulation, we did some extensions and implemented the Extended Randomized Protocol on a real test bed. The Extended Randomized Protocol is combined with a randomized routing, randomized MAC, a contention scheme, frequency division access scheme and adaptive sleeping disciple, which allow to satisfy constraints on latency and error rate, while optimizing for energy consumption. We characterized the protocol performance with a mathematical model that allowed to set the working point of nodes by solving a constrained optimization problem. In addition, we studied a completely distributed algorithm that allows for the network to reach the optimal working point and adapt to traffic variations and channel condition changes.

Experimental results presented in this thesis report demonstrate the effectiveness of the implementation and show how our protocol can support different traffic rate and channel conditions with Tmote Sky sensor in a indoor environment. Although the mathematical model used is not complex, it allows the protocol to adapt the changes in traffic rate and channel condition. Since our protocol provides low E2E delay, it can be applied to real-time control in industrial automation.
From own results, we can conclude that extra wake up rates can deteriorate performance of the network. Although higher wake up is necessary for better packet reception rate, it causes higher energy consumption, duplicate packets

and extra delays. The main problem of Extended Randomized Protocol is the bad working of transition period. If the transition period problem is overcome, the accuracy of the system will increase significantly.
However, given experimental results, Extended Randomized Protocol is a reliable protocol that is capable of achieving the required error rate and E2E delay with providing the load balancing in sensor network.

6.2 Summary of contribution and achievements

During the development of Extended Randomized Protocol, different contributions and achievements were obtained, these are:

- Extended Randomized Protocol contributes in making further steps in the implementation of MAC layer, routing protocol and power control at the Automatic Control Lab, school of electrical engineering at the Royal Institute of Technology in Stockholm.
- Description of the Extended Randomized Protocol is provided. This description includes state diagrams, documentation, implementation code and experimental results presenting the capabilities and performance of the implemented cross-layer protocol.
- Solid knowledge about WSNs and MAC protocols for WSNs was acquired during the development of this work. This knowledge will help future research related to the implementation of approaches and improvement of theory for different WSN applications.
- The description of Extended Randomized Protocol and its implementation makes a contribution in the field of MAC layer, routing protocol and power control.

6.3 Future work

This section pursues the identification of further points for research and development in order to improve performance of Extended Randomized Protocol. For improvement of general performance, it is necessary to offer longer network lifetime , QoS requirement and so on. Therefore, the following points of reference for further work are identified.

- Message loss

  Message loss in the sensor network is a problem that needs to be looked at. The current Extended Randomized Protocol needs to be improved to prevent the high message loss rates when nodes are within communication range of each other.

- Optimal transition period

  Larger tests or a mathematical model is needed to determine the optimal transition period to the change the number of clusters in the network.

- Duplicate packets

  Duplicate packets cause the higher traffic load and extra energy consumption. Although we implement the contention scheme to avoid the duplicate packets, the problem is not completely solved.

- Network processing

  Data aggregation [34] performs in-network fusion of data packets, coming from different sensors enroute to the base station, in an attempt to minimize the number and size of data transmissions and thus saving sensor energies.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] K. Römer, O. Kasten, and F. Mattern, "Middleware challenges for wireless sensor networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 6, no. 4, pp. 59–61, 2002.

[3] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[4] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[5] M. Tubaishat and S. Madria, "Sensor networks: an overview," *Potentials, IEEE*, vol. 22, pp. 20–23, 2003.

[6] L. van Hoesel, T. Nieberg, J. Wu, and P. J. M. Havinga, "Prolonging the lifetime of wireless sensor networks by cross-layer interaction," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 78 – 86, 2004.

[7] Q. Jiang and D. Manivannan, "Routing techniques in wireless sensor networks: a survey," *CCNC '04*, pp. 93 – 98, 2004.

[8] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE, Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[9] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM TRANS*, vol. 12, no. 3, pp. 493 – 506, 2004.

[10] W. Ye and J. Heidemann, "Medium access control in wireless sensor networks," in *Wireless sensor networks*. Norwell, MA, USA: Kluwer Academic Publishers, 2004, pp. 73 – 91.

[11] T. V. Dam and K. Langendoen, "An adaptive energyefficient mac protocol for wireless sensor networks," in *SenSys '03*. New York, NY, USA: ACM Press, 2003, pp. 171–180.

[12] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *SenSys '04*. New York, NY, USA: ACM Press, 2004, pp. 95 – 107.

[13] I. Rhee, A. Warrier, M. Aia, and J. Min, "Zmac: a hybrid mac for wireless sensor networks," in *SenSys '05*. New York, NY, USA: ACM Press, 2005, pp. 90 – 101.

[14] A. Bonivento, C. Fischione, and A. Sangiovanni-Vincentelli, "Randomized protocol stack for ubiquitous networks in indoor environment," *CCNC*, vol. 1, pp. 152– 156, 2006.

[15] J. V. Greuen, D. Petrovic, A. Bonivento, J. Rabaey, K. Ramchandran, and A. Sangiovanni-Vincentelli, "Adaptive sleep discipline for energy conservation and robustness in dense sensor networks," *ICC*, vol. 6, pp. 3657 – 3662, 2004.

[16] C. Dazhi, C. Guangtong, and Z. Long, "A multihop data relay scheme for wireless networked sensors," *VTC '05*, vol. 3, pp. 1814–1818, 2005.

[17] A. Bonivento, C. Fischione, A. Sangiovanni-VIncentelli, F. Graziosi, and F. Santucci, "Seran: A semi random protocol solution for clustered wireless sensor networks," *MASS '05*, November 2005.

[18] R. Verdone and C. Buratti, "Modelling for wireless sensor network protocol design," *IWWAN*, 2005.

[19] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *System Sciences*, vol. 2, p. 10, 2000.

[20] *Tmote Sky Data Sheet*, Moteiv, San Francisco, CA, 2006. [Online]. Available: http://www.moteiv.com/products/docs/tmote-sky-datasheet.pdf

[21] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," *INFOCOM*, vol. 3, no. 4, pp. 366 – 379, 2004.

[22] *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Computer Society Std. 802.15.4, 2003.

[23] C. Won, J.-H. Youn, H. Ali, H. Sharif, and J. Deogun, "Adaptive radio channel allocation for supporting coexistence of 802.15.4 and 802.11b," *VTC '05*, vol. 4, pp. 2522– 2526, 2005.

[24] *CC2420 Data Sheet*, Chipcon, Oslo, Norway, 2005. [Online]. Available: http://www.chipcon.com/files/CC2420-Data-Sheet-1-3.pdf

[25] L. Ahlin and J. Zander, *Principles of Wireless Communication*. Lund, Sweden: Studentlitteratur, 2006.

[26] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," *IEEE SECON'04*, pp. 517– 526, 2004.

[27] L. D. Paolo, C. Fischione, F. Graziosi, F. Santucci, and S. Tennina, "Performance analysis of distributed source coding and packet aggregation in wireless sensor networks," in *Proc. IEEE Globecom '06*, 2006.

[28] K. Srinivasan† and P. Levis, "Rssi is under appreciated," in *EmNets 2006*, 2006.

[29] "Research integration: Platform survey," White Paper, European Commission, 2006.

[30] *Moteiv : Accelerating Sensor Neworking*, Moteiv, San Francisco, CA, 2003. [Online]. Available: http://www.moteiv.com/

[31] D. Gay, P. Levis, and D. Culler, "Software design patterns for tinyos," in *Proc LCTES '05*, vol. 31, no. 15. New York, NY, USA: ACM Press, 2005, pp. 40–49.

[32] The tinyos community forum. [Online]. Available: http://www.tinyos.net

[33] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *SIGOPS Oper. Syst. Rev.*, vol. 34, no. 5, pp. 93–104, 2000.

[34] S. Lindsey and C. S. Raghavendra, "Power efficient data gathering and aggregation in wireless sensor networks," *SIGMOD Rec.*, vol. 32, no. 4, pp. 66 – 71, 2003.

Appendix A

Packet structure

- **TinyOS Message**

```
typedef struct TOS_Msg{
    uint8_t length;      \\ Length of the payload
    uint8_t fcfhi;       \\ Frame control byes
    uint8_t fcfho;       \\ Frame control byes
    uint8_t dsn;         \\ Frame control byes
    uint16_t destpan;    \\ Destination identifier
    uint16_t addr;       \\ Destination address
    uint8_t type;        \\ Message type
    uint8_t group;       \\ Group ID
    uint8_t data[28];    \\ Data packet up to 28 bytes
} TOS_Msg
```

- **Uart Message**

```
AM_UARTMSG = 30                \\ Uart message type
typedef struct UartMsg{
    uint8_t trafficLambda;     \\Traffic rate λ
    uint16_t sourceMoteID;     \\ Last intermediate node's ID
    uint32_t sourceTime;       \\ Packet generated time in Source node
    uint32_t destinationTime;  \\ Receiving time in Destination node
    bool AIMD;                 \\ AIMD command
    uint16_t seq;              \\ Packet sequence number
    uint8_t channel;           \\ Successful packet reception rate
    uint8_t period_channel;    \\ Channel condition p in fixed period
    uint8_t blocks;            \\ Number of clusters in network
    uint16_t time;             \\ time in Destination node
} UartMsg_t
```

- **Data Message**

```
AM_DATAMSG = 10            \\ Data message type
typedef struct DataMsg{
    uint8_t trafficLambda; \\ Traffic rate λ
```

```
    uint16_t sourceMoteID;\\ Intermediate node's ID who sent data msg
    uint32_t sourceTime;  \\ Packet generated time in Source node
    bool AIMD;            \\ AIMD command
    uint16_t seq;         \\ Packet sequence number
} DataMsg_t
```

- **Synchronization Message**

```
AM_DATAMSG = 15           \\ Synchronization message type

typedef struct SyncMsg{
    uint16_t sourceMoteID;\\ Node's ID who sent synchronization msg
    uint32_t globalTime;  \\ Global time to synchronize the time
} SyncMsg_t
```

- **Beacon Message**

```
AM_BEACON = 20                \\ Beacon message type

typedef struct BeaconMsg{
    uint8_t channel;          \\ Channel condition p
} BeaconMsg_t
```

Appendix B

Experimental Parameters

Table B.1: Experimental parameters

Parameter	Value
`MAX_SEND_TRIES` : Maximum send tries in B-MAC layer	5
Number of intermediate nodes	15
`DISTANCE` : Network size	$20m$
β : Path loss expoent	2.8
$PL(d_0)$: Path loss for reference distance d_0	$55dB$
d_0 : Reference distance	$1m$
P_{ni} : Node floor and interference	$-115dBm$
σ : Standard deviation	6.8
$\Theta_{max,s}$: Maximum required number of packets to estimate a first channel condition	800
ψ_{max} : Maximum required number of packets to compute the channel estimation period	100
`beacon_freq` : Beacon communication channel	26
`msg_freq` : Data communication channel	25
`sync_freq` : Synchronization channel	11
δ : Constant offset value for initial channel condition	0.05
τ : E2E delay constraint	$5s$
Δ : Addictive Increase command in AIMD	3
ψ : Multiplicative Decrease in AIMD	1.2
T_{dmax} : Maximum delay of contention scheme	$12ms$
T_d : Random delay of contention scheme	$0, 3, 6, 9, 12ms$
T_a : Maximum active time to receive data message	$30ms$
T_w : Maximum waiting time to receive beacon message	$1s$
μ_{init} : Initial wake up rate per second	3
l_m : Length of data message	$28bytes$
l_b : Length of beacon message	$19bytes$
E_{Te} : Energy consumption on Tx mode per bit	$234.0nJ/bit$
E_{Re} : Energy consumption on Rx mode per bit	$261.6nJ/bit$
F : Time containing propagation and transmission of data message	$0.875ms$
E_{ac} : Energy consumption in active time T_ac	$1962\mu J$
E_w : Energy consumption in contention time T_w	$392.4\mu J$
ϵ_m : Energy consumption to transmit data message a distance $1m$ per bit	$31.7952\mu J$
ϵ_b : Energy consumption to transmit beacon message a distance $1m$ per bit	$12.7680\mu J$