ENERGY-EFFICIENT PROTOCOLS AND TOPOLOGIES FOR SENSOR AND PERSONAL-AREA NETWORKS

By

YUANYUAN ZHOU

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY
School of Electrical Engineering and Computer Science

AUGUST 2007

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of YUANYUAN ZHOU find it satisfactory and recommend that it be accepted.

_____
Chair

_____

_____

_____

ACKNOWLEDGEMENT

First, I would like to acknowledge my advisor, Dr. Murali Medidi. He is my mentor not only in career, but also in life. Without him, this dissertation would not have been possible. I thank him for his patience and encouragement that carried me on through difficult times, and for his insights and suggestions that helped to shape my research skills.

I would like to thank other members on my defense committee including Dr. Sirisha Medidi, Dr. K. C. Wang, and Dr. Zhe Dang. Their valuable feedback helped me to improve the dissertation in many ways.

I have been working in SWAN Lab for three years, during which I obtained a lot of help and valuable research experience. I feel really lucky to be a member of this group and would like to thank all the members in the group, including Christopher J. Mallery, Peter M. Cappetto, Monique Kohagura, Ghayatri Garudapuram, and Jiong Wang. I would also like to thank Monique for her great food and snack that make me energetic to conduct research.

Last but not least, I thank my wife and my parents for their love and support through all these years. They are always my most powerful spirits for moving forward.

ENERGY-EFFICIENT PROTOCOLS AND TOPOLOGIES FOR SENSOR AND PERSONAL-AREA NETWORKS

Abstract

by Yuanyuan Zhou, Ph.D.
Washington State University
AUGUST 2007

Chair: Muralidhar Medidi

We study energy-efficient topologies and protocols in Wireless Sensor Networks (WSN) and Personal Area Networks (PAN). In both networks, energy-efficiency is the primary objective for designing communication protocols since the deployed devices are usually battery-powered and resource-constrained. We also identified network performance requirements (e.g., end-to-end delay, throughput, etc.), and improve energy-efficiency without sacrificing network performance. We identify WSN's distinct requirements on the Medium Access Control (MAC) protocol and trade-offs among delay, energy, and throughput. Based on these requirements and trade-offs, we propose ECR-MAC (for *Energy-efficient Contention-Resilient MAC*) which employs a *Dynamic Forwarder Selection* technique to improve both energy-efficiency and delay without requiring additional synchronization or radio hardware support, and efficiently handle spatially-correlated contention. Besides conserving independent node's energy in ECR-MAC, we also study the "energy hole" problem in WSN, and propose a differential duty cycle assigning approach to balance energy consumption overall the network. Results show that ECR-MAC provides significant energy-savings, low delay and high network throughput, and our differential duty cycle assigning approach can further improve network lifetime without sacrificing network performance. To provide a short bounded end-to-end delay for WSN's real-time sensor applications that may not be satisfied by ECR-MAC, we propose a sleep-based topology control technique, which facilitates to implement an efficient wakeup scheduling method by leveraging time synchronization and localization. Through analysis and simulations, we show that our topology control and wakeup scheduling can provide the end-to-end delay that is generally bounded by a small factor

to a fully-active approach, achieve higher throughput than ECR-MAC, and conserve more energy. To address energy-efficiency problems in personal area networks, we propose a multi-hop energy-aware scatternet (EMTS) formation technique for Bluetooth-based PAN, and further design a local reorganization approach that maintains scatternets' energy-efficient features to extend scatternet lifetime. Our scatternet formation technique balances devices' energy consumption by assigning roles to devices that suit their workload and energy resources, and efficiently form the scatternets with short links and small network diameter to reduce energy consumption. The lightweight reorganization can extend scatternet lifetime further. Experimental results confirm that the generated scatternets have a significantly longer lifetime than others.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Dedication

I dedicate my dissertation to my wife Yue, my parents
Rongzhong and Xiaoming, and my little sister Yiru, who
were always there for me throughout the entire doctorate program.

CHAPTER ONE

INTRODUCTION

1.1 Overview

With the advances in microelectronic fabrication, processor capability, and wireless communications, computing devices have evolved from huge and centralized workstations to distributed portable devices, such as smart phones, notebooks, personal digital assistants, and even tiny sensors. It is desired to interconnect these devices through wireless links due to their portable nature, and there is a promising development of networking technologies for interconnecting these devices, e.g., wireless sensor networks (WSN) and personal area networks (PAN).

A WSN is a distributed sensing network comprised of a large number of small sensor nodes that detect and report information about the environment [9]. Each sensor node, besides being equipped with a sensor that can detect certain physical phenomenon (e.g., temperature, light, humidity, acoustics, etc.), also has a radio that can send/receive packets. To collect sensing data, there exist one or a few *base stations* that gather and process sensors' reports, and eventually provide them to users. Sensor nodes are expected to be inexpensive and deployed in a large numbers, so compared to traditional devices in computer networks, sensor nodes have very limited energy, processing power, storage, communication range and rate. For example, the Mica2 Mote developed by UC Berkeley is typically powered by two AA batteries and uses a low-power micro-controller: 7.3 MHz Atmega 128L processor [4]; its memory is less than 1MB (128KB of code memory, 512KB EEPROM, 4KB of data memory); and it uses ChipCon CC1000 radio with radio rate of only 38.4Kbps and range of approximately 300m. WSN have been widely deployed in residential, commercial, medical, industrial, and military applications, such as target tracking [6], habitat sensing [16, 44] and fire detection.

A Personal Area Network (PAN) is the interconnection of computing devices within the range of an individual person. The network area is usually limited, typically within 10 meters. For instance, a person can carry a laptop, and travel with a personal digital assistant (PDA), a digital camera, and a portable printer. All of these computing devices can be interconnected and form a PAN without having to plug anything in. Due to the portable nature of these devices, PAN is virtually a synonym of wireless personal area network

(WPAN) since almost any PAN would need to function wirelessly.

Since most sensor nodes and portable devices are energy-constrained, a common feature of sensor nodes and PAN devices, as well as their networks, is the requirement for energy-efficiency. Sensor nodes are usually unattended once deployed, and it is expected that a WSN should sustain for years without human operation. A desirable PAN should also avoid to recharging devices frequently. Researchers have conducted intensive works on conserving energy and extending lifetime of WSN and PAN. A widely-used energy-conserving technique is to use active/sleep duty cycles, i.e., turn off devices' radios periodically. For instance, Bluetooth, which has been regarded as an enabling technology for PAN, allows to deploy duty cycles in which each device only needs to be active during its preassigned communication time slots. Similar duty cycles have been widely used in WSN to reduce the energy consumed by idle listening. Another popular-used technique is to minimize radios' transmission power, i.e., using short links for device communication.

Besides energy-efficiency, WSN and PAN naturally have several other important performance metrics that should be satisfied to improve their applicabilities. Generally, short end-to-end delay, high throughput, and low interference are always desirable. However, these performance metrics usually create trade-offs to the energy-efficiency. In this dissertation, our focus is to answer the following question: *how to improve network's energy-efficiency while guaranteeing network's communication performance*? We explore this issue by proposing new protocols and efficient topologies for WSN and PAN.

1.2 Wireless Sensor Networks

1.2.1 Medium Access Control (MAC)

To effectively support sensor applications, each node in a WSN, like in traditional wireless networks, needs to implement a network stack that includes network protocols in various layers (e.g., *Medium Access Control* (MAC) layer, routing layer, transport layer, etc.). As a basic communication building block, MAC layer has significant effect on network performance and energy-efficiency. The objective of MAC layer is to coordinate devices' access to a common communication channel, so as to minimize packet collisions, shorten end-to-end delay, and improve network throughput. MAC layer also controls turning on/off radios, which directly influences devices' energy-efficiency. Due to sensor nodes' restricted resource and WSN's specific

organization pattern, WSN has the following distinct features that should be considered when designing its MAC protocols.

- Energy-Efficiency: energy-efficiency is usually regarded as the primary designing objective for WSN MAC protocols [9, 34]. Essentially, sensor nodes require two core functionalities: computation and communication. Previous researches have shown that communication energy consumption is definitely comparable and even outweighs that of computation [67]. Furthermore, the energy consumption of computation is expected to decrease due to Moore's law and rapid advance of digital circuit designing in the future, while the energy consumption of communication is physically determined by Maxwell's laws and does not follow the same trend as that of computation. Therefore, our works focus on energy-efficient approaches to conserve nodes' communication energy. In the following, we simply use *energy consumption* to denote the energy consumed by communication (i.e., radios).

- Low-Latency: MAC protocols that achieve short end-to-end delay are preferred especially for surveillance sensor applications [42, 23, 60]. For instance, a sensor application that monitors forest fires may require fire alarm messages to be received within a short bounded time period to enable a quick response.

- Contention-Resilience: WSNs are expected to be deployed with a high density, i.e., multiple sensor nodes may be deployed in the same geographical area for fault-tolerance. Further, the traffic on WSN appears in a bursty fashion when certain events happen, which will lead to the "spatially-correlated contention" [28, 11], i.e., several close-by nodes that sense the same event are sending reports at almost the same time. Without careful designing and controlling, spatially-correlated contentions can cause extensive packet collisions and greatly degrade network performance. WSN MAC protocols should be designed to handle contention gracefully.

- Simple Implementation: WSN MAC protocols should be simple with low overhead due to nodes' constrained resources.

- Scalability: a typical WSN is large and dense, so WSN MAC protocols should scale with high density and a large numbers of nodes [34, 58].

Compared to traditional wireless networks, WSN's distinct requirements make network protocol designing more challenging. On the other hand, WSN also has a few features that can be exploited to facilitate protocol designing. First, sensor nodes usually collaborative for a common sensing task, so the fairness among independent sensors is not critical for typical sensor applications [87]. Second, since base stations are responsible for collecting sensor reports, there exists a predominant many-to-one traffic pattern: *convergecast*. Convergecast reduces the number of possible traffic in a WSN compared to the all-to-all traffic pattern, but it also introduces the spatially-correlated contention that needs to carefully addressed. Third, sensor nodes are expected to be static all their lifetime since they are typically unattended once deployed, which simplifies protocol design and eliminates mobility issues for most scenarios. Fourth, sensor nodes typically have similar computation and communication capability, so a WSN can be regarded as a homogenous network that helps to simplify protocol design. Finally, for most sensor applications, sensor nodes do not communicate frequently – they send reporting packets in an event-driven manner [42]. So it is not necessary to turn on nodes' radios all the time.

Since most WSN nodes only spend a small portion of time reporting sporadic events, active/sleep duty cycles (i.e., nodes turn on their radios periodically and sleep for the rest time) have been widely used by WSN MAC protocols to reduce energy wastage caused by idle-listening. Duty cycle is defined as $T_{active}/(T_{sleep}+T_{active})$ [67], where T_{active} denotes the active period and T_{sleep} denotes the sleep period. To maximize the energy-efficiency, it is desirable to minimize the duty cycle, i.e., minimize T_{active} and maximize T_{sleep}. However, there exists a trade-off between energy-efficiency and delay since each sender needs to spend an additional setup latency T_{setup} to wake up its next-hop forwarder [67].

Earlier works in designing MAC protocol generally achieved energy-efficiency with the cost of sacrificing network performance (e.g., enlarging delay and decreasing throughput) [67, 88]. Later several MAC protocols were proposed to improve both energy-efficiency and end-to-end delay. Most of them require network-wide synchronization [42, 23] and/or dual radio setup [67, 85], which increases hardware cost and protocol overhead. It is desirable to design a WSN MAC protocol that improves network performance with low overhead and handles the trade-offs among different requirements in an integrated and balanced manner.

1.2.2 Energy Hole Problem

Convergecast, the predominant traffic pattern for typical sensor applications, requires the nodes closer to the base station to forward packets for the nodes that are farther away, and hence the nodes around the base station carry heavier traffic load and deplete their energy more quickly. This phenomenon is termed *energy hole* [36, 50], which causes unbalanced energy consumption among sensor nodes and impairs network lifetime. The energy hole problem has attracted extensive interest in the past few years, and various approaches have been proposed to address it [36, 50, 82].

Essentially the energy hole is caused by the specific traffic pattern in a WSN, and most approaches that address this problem assume the traffic as the only source of energy consumption [36, 82]. However, in typical sensor applications, the energy consumed by idle listening can not be ignored [67, 43], and duty cycles have been widely utilized to save energy wastage caused by idle listening. Existing protocols with duty cycles, which assign all nodes the same duty cycle, did not consider the energy hole, so nodes closer to the base station can still drain their energy quicker and impair the lifetime. To maximize network lifetime, it is desirable to combine the duty cycles with approaches of addressing the energy hole problem, which not only reduces the energy wastage of idle listening, but also balances the energy consumption across the network, and hence extends network lifetime.

1.2.3 Topology Control

Topology control is an effective method to conserve energy and improve network performance in WSN. From the perspective of functionality, topology control aims at providing a suitable topology (made up of nodes and communication links) from the original network to facilitate network communication. Typically, topology control techniques are designed based on nodes' features (e.g., residual energy, active/sleep pattern, etc.), position, and distances between each other, such that the generated topology helps to enhance network performance, e.g., better energy-efficiency, extended network lifetime, shorter end-to-end delay, and lower interference, etc. [66].

There exist several approaches to control networks' topologies: topology control techniques in traditional wireless networks usually assume nodes are active all the time, so they achieve energy-efficiency by

minimizing the transmission power and maintaining the minimal-energy path among any two nodes. However, in most sensor applications, nodes do not communicate frequently, so the most effective energy-saving method is to turn off nodes' radios as much as possible, i.e., using low active/sleep duty cycles [67, 85]. By following this direction, sleep-based topology control has been proposed to improve energy-efficiency and performance of WSN.

Sleep-based topology control utilizes space redundancy and duty cycles to conserve energy. It selects a subset of nodes for constructing the backbone and communicating, and the remaining nodes turn off their radios and periodically check their eligibility to replace backbone-nodes according to certain performance metrics [84, 18]. So each node generally follows an active/sleep duty cycle and hence extends the network lifetime compared to a fully-active protocol (e.g., IEEE 802.11 [7]). The utilization of duty cycles introduces new challenges for topology control techniques. First, the transmission power minimization is no longer necessary for energy-saving especially considering that the power adjustment requires more advanced hardware equipments that may not be available in tiny sensor nodes. Second, the end-to-end delay is significantly impacted by the setup latency, so the topology control techniques that bound the route length (in hops) no longer necessarily leads to a delay reduction [26].

To sustain network performance (i.e., delay, throughput), most sleep-based topology control techniques ensure retaining a connected network at any time. However, for a sensor application, keeping a network connected all the time is not always necessary. Considering a forest fire monitor system: ideally the network is only required to be connected when sending critical fire alarm reports, which happens very infrequently. So a potential solution that can achieve better energy-efficiency is to allow a WSN to be disconnected temporarily and put nodes into sleep mode as much as possible. This solution is more attractive for event-driven sensor applications where events rarely happen and the whole network is expected to sustain as long as possible. By following this direction, researchers have proposed several wakeup scheduling approaches which use a much lower duty cycle and coordinate nodes' wakeup times to facilitate communications. These wakeup scheduling approaches typically conserve more energy by allowing network to become temporarily disconnected. So an issue worthy of exploring for wakeup scheduling is: *how to maintain the network performance (e.g., delay, throughput) even if network is not guaranteed to be connected all the time (i.e., with low duty cycles)*? Implementing efficient wakeup scheduling requires to combine topology control

techniques and nodes' wakeup coodinations. We will explore sleep-based topology control techniques to support wakeup scheduling that achieves better energy-efficiency while maintaining network performance.

1.3 Energy-Aware Personal Area Networks

Similar to a WSN, the topology of a PAN has a significant effect on the energy-efficiency and communication performance. However, compared to a WSN, a PAN has the following distinct features which should be considered for constructing PAN topologies.

- Heterogeneous Networks: unlike a WSN that is typically made up of identical sensor nodes, a PAN usually contains various kind of devices in terms of energy capability, processing power, storage, etc. For instance, a laptop has a larger energy capacity and better computing capability than a PDA, which in turn is more powerful than a sensor node. Techniques forming topologies for a PAN should leverage devices' various features to accomplish performance improvement and energy-efficiency.

- Adjustable Transmission Power: most PAN devices (e.g., laptop, PDA, etc.) are usually equipped with more advanced and sophisticated hardware than sensor nodes, which enables a device to control its transmission range and avoid unnecessary energy wastage when communicate to a nearby device.

- All-to-all Traffic Pattern: in a PAN, each device typically acts as an independent communication component, so all-to-all traffics pattern is prevalent in a PAN, which is different from a WSN where convergecast is the predominant traffic pattern. These two different traffic patterns can significantly affect the topology designed for PAN and WSN.

- Device Mobility: mobility is a nature of portable devices, so topologies for a PAN should accommodate to dynamic environment as opposed to WSN that are expected to be static throughout lifetime.

As a low-cost, low-power short range wireless communication technology, Bluetooth is considered an enabling technology for PAN construction [22]. Bluetooth devices can form small networks called piconets. In any piconet, there is precisely one master and the number of active slaves is bounded by 7. Piconets can be interconnected to form larger networks, called *scatternets*, by sharing devices, termed *bridges*. While the Bluetooth specification supports the notion of a scatternet formation, it does not specify either a topology or its formation. The problem of scatternet formation can be regarded as assigning roles (i.e., master, slave,

and bridge) to Bluetooth devices and establishing links between devices, and the topology of the generated scatternet greatly affects the overall network performance, especially the devices' energy-efficiency and network lifetime [29, 52]. So it is desirable to form an energy-aware scatternet to improve its sustainability. Due to PAN's distinct features, the topology control techniques proposed for a WSN can not be directly applied for a scatternet. To determine and construct an optimal energy-aware scatternet is an extremely complex problem due to the large number of parameters involved, e.g., device roles, link length, diameter, etc. Marsan *et al.* have shown a centralized algorithm to determine the optimal scatternet that minimizes the energy consumption of the most congested device in the network [45]. However, their centralized algorithm is shown to be NP-complete, which is too expensive to be adopted for distributed Bluetooth devices. Therefore, "heuristics for the construction of good (suboptimal) topologies in a distributed fashion are needed" [45].

Scatternet maintenance is another important issue that needs to be addressed to retain energy-efficiency and handle device mobility, since varying workload across devices may deplete some devices more quickly than others and hence impair the network lifetime, and devices' departure and join should be efficiently handled to avoid network disconnection and communication interruption. Naturally, the maintenance overhead should be minimized to ensure a scatternet's communication performance.

1.4 Research Contributions and Outline

Our research efforts are targeted toward energy-efficient protocols and topologies in WSN and PAN. For WSN, we explored energy-efficient MAC protocol design as well as addressed the energy hole problem, and we also proposed topology control techniques to improve network performance; for PAN, we studied scatternet formation and maintaining problems. In particular, we have the following research contributions:

1.4.1 Energy-efficient MAC for WSN

In Chapter 2, we proposed a new MAC protocol: *Energy-efficient Contention-Resilient MAC* (ECR-MAC). It employs sensor node redundancy to improve both energy-efficiency and end-to-end delay without involving additional hardware cost or synchronization overhead. ECR-MAC addresses the spatially-correlated contention in sensor applications by diffusing traffic, which effectively improves network throughput and minimizes energy wastage. Besides performing simulations to evaluate ECR-MAC's performance, we have

implemented ECR-MAC using MICA2 motes test-bed. Experiment results show that ECR-MAC achieves shorter end-to-end delay, higher throughput and saves more energy in both single-source and multiple-source scenarios with spatially-correlated contention, and it scales with the network density very well.

As a further study in WSN MAC protocols, we analyzed nodes' energy consumption by both communication traffic and idle listening in a WSN, and addressed the energy hole problem in ECR-MAC by assigning different duty cycles to nodes with different distances from the base station. This differential approach not only reduces the energy wastage of idle listening, but also balances the energy consumption across the network, which leads to significant lifetime extension (by up to about 50%) compared to uniform duty cycles without sacrificing network performance (e.g., end-to-end delay and throughput).

1.4.2 Sleep-based Topology Control in WSN

ECR-MAC is designed with the aim to improve energy-efficiency and network performance (e.g., delay, throughput) with low overhead, i.e., no synchronization requirement. Though synchronization increases protocol overhead, it enables to achieve shorter delay bound by coordinating nodes' wakeup times. In ECR-MAC, nodes choose their wakeup times independently, which brings the benefit of simplicity and efficiency, but can lead to a high worst-case end-to-end delay since it does not leverage synchronization to coordinate nodes' wakeup times. To provide better support for real time sensor applications that require short and bounded end-to-end delay, in Chapter 3 we proposed a sleep-based topology control technique to support an efficient wakeup scheduling that can achieve the end-to-end delay bounded by a small factor of the one achieved by a fully-active protocol (i.e., without using duty cycles). Both analytical and simulation results showed that our topology control can lead to significant energy-savings, short bounded end-to-end delay, and high throughput under spatially-correlated contention.

1.4.3 Energy-aware Bluetooth Scatternets for PAN

In Chapter 4, we proposed a distributed technique to form energy-aware scatternets, The proposed scatternet formation technique balances devices' energy consumption by assigning roles to devices that suit their workload and energy resources, and efficiently form scatternets with short links and small network diameter to reduce energy consumption. The generated scatternets are also designed for a small number of piconets

and a low device degree to improve communication performance in typical Bluetooth applications. A localized, and hence lightweight, reorganization technique was proposed to extend scatternet lifetime further. Experimental results, obtained from implementations using the Blueware simulator [2], confirm that the generated scatternets have a significantly longer lifetime than others.

CHAPTER TWO

ENERGY-EFFICIENT CONTENTION-RESILIENT MEDIUM ACCESS FOR WIRELESS SENSOR NETWORKS

2.1 Energy-efficient Requirements in Wireless Sensor Networks

Energy-efficiency has been regarded as the primary designing objective for WSN MAC protocols, and the potential expectation of an energy-efficient WSN is that it can sustain as long as possible, i.e., achieve a long network lifetime. The definition of network lifetime may depend on the application scenario in which the network is used. However, it is widely believed that node failures can severely degrade network performance and even disconnect the network, so prior works usually define it as the time till the first node depletes its energy [52, 70]. To extend network lifetime, there are two possible approaches that are compensated to each other: minimizing each node's energy consumption and balancing nodes' energy consumption across the whole network.

2.1.1 Minimize Energy Consumption

Minimizing independent node's energy consumption is obviously an effective method for achieving energy-efficiency and lifetime extension. In [88], Ye. *et al.* identified four possible sources of energy wastage in WSN MAC protocols:

- Collision: a packet collision causes the receiver to receive a corrupted packet and need to receive for a second time; senders need to retransmit packets, all of which increases energy wastage.
- Overhearing: when a node turns on its radio unnecessarily, it can hear packets destined to other nodes.
- Control packet overhead: in order to coordinate channel access and confirm packet transmission, WSN MAC protocols include various kind of control packets, which, though is usually unavoidable, causes energy wastage for data transmission.
- Idle listening: nodes turn on their radios to wait for possible communications without receiving any data.

For a typical WSN application, communication happens infrequently, so idle listening has been regarded as the major energy wastage source. In the literature, duty cycle has been show to be an effective mechanism for reducing idle listening and been widely used in WSN MAC protocols [88, 74, 42]. However, since a node that turns off its radio can not participate in network communication, applying duty cycles can adversely affect network performance (e.g, delay, throughput), which brings the following trade-offs to MAC protocols.

- Energy-delay trade-off: when a node turn off its radio, it can not be used as a forwarder until it turns on radio again. So any node that attempts to use this node for forwarding packets needs to spend a *setup latency* (i.e., the delay to wait a forwarder to wake up), which obviously prolongs end-to-end delay. In general, the lower the duty cycle (and hence the lower energy consumption), the higher the delay.

- Delay-throughput trade-off: this trade-off seems counter-intuitive since usually shortening delay helps to improve throughput. However, this trade-off does exist due to contention and the deployment of duty cycles. For instance, to reduce the setup latency and hence shorten end-to-end delay, researchers have proposed a *staggered* wakeup scheduling method that sequentially coordinates wakeup times of the nodes in a packet transmission path, and the whole network is organized as convergecast tree rooted at the base station [42, 23]. This method, though is effective to reduce delay, requires nodes at the same tree level to wakeup at the same time, which can causes intensive collisions under contention and impair throughput.

- Trade-off between performance and protocol simplicity: in recent years, researchers have proposed various methods to improve both energy-efficiency and delay. However, most of them require network-wide synchronization and/or dual radio setup. Synchronization increases protocol overhead especially considering that it needs to be performed periodically to overcome time drifts. Dual radio setup increases the radio hardware cost and may not be available in tiny sensor nodes.

A desirable WSN MAC protocol should address the above-mentioned trade-off in an integrated and balanced manner, so as to minimize nodes' energy consumption as well as improve network performance with low overhead.

2.1.2 Balance Energy Consumption

Balancing energy consumption, compared to minimizing energy consumption, operates in an orthogonal dimension and can also effectively improve network lifetime. From the balance perspective, the ideal case is that all the nodes in the network deplete their battery power at the same time, such that no residual energy is wasted after the lifetime expires. However, WSN's specific traffic pattern convergecast typically causes skewed energy consumption across the network.

Convergecast requires the nodes closer to the base station to forward packets for the nodes that are farther away, and hence the nodes around the base station carry heavier traffic load and deplete their energy more quickly. This phenomenon is termed *energy hole* [36, 50], which causes unbalanced energy consumption among sensor nodes and impairs network lifetime. Energy hole problem has attracted extensive interest in the past few years, and various approaches have been proposed to address it [36, 50, 82]. Most approaches that address this problem assume the traffic as the only source of energy consumption [36, 82], while ignoring the energy consumed by idle listening, which can significantly impair their effectiveness since they can not co-exist with the widely-used duty cycles that achieve significant energy savings.

To make our energy-balancing approach compensate to duty cycles that minimizes energy consumption and hence reinforce energy-efficiency, we design a *differential* duty cycle assigning approach based on the analysis of the energy consumed by both traffics and idle listening. This approach can be applied in most WSN MAC protocols that use duty cycles, as long as they are not critically dependent on all nodes using the same duty cycle (e.g., synchronized staggering wakeup mechanisms [42]). So compared to the methods that merely minimize nodes' energy consumption through duty cycles, our differential approach can further improve network lifetime by addressing the energy hole problem.

Similar to the issues in minimizing energy consumption, addressing the energy hole problem should also consider possible effects on the network performance. Since most sensor applications require data reports to reach the base station quickly to enable a timely response [20], end-to-end delay is a critical performance metric that has significant effect on sensor applications. However, existing approaches that address the energy hole problem have not considered these impacts on the end-to-end delay, which can lead to undesirable network performance degradation for achieving lifetime extension. Our differential approach is designed to address energy hole problem without sacrificing network performance, e.g., end-to-end delay

and throughput.

2.2 Related Work

2.2.1 MAC Protocols in WSN

In the last few years, a lot of MAC protocols have been proposed for WSN. They were designed based on different assumptions about the network and had different focuses on the network performance metrics. In the following, we will briefly introduce them according to their channel access mechanisms: CSMA (*Carrier Sense Multiple Access*), TDMA (*Time Division Multiple Access*), and a hybrid between these two.

CSMA MAC Protocols

CSMA MAC allow nodes to access the medium at any time without previous coordinations, which simplifies protocol implementation and brings flexibility to accommodate mobile nodes and traffic fluctuations [67, 23, 28, 85, 7]. The major disadvantages of CSMA protocols are the possible collisions that decrease bandwidth efficiency and cause energy wastage. To save energy, most CSMA protocols for WSN have employed active/sleep duty cycles to reduce the idle listening period [23, 67, 85].

STEM is one of the first CSMA-based MAC protocols for WSN that utilizes duty cycles [67]. STEM deploys two radios that operate in different channels: a wakeup channel and a data channel. To extend the network lifetime, STEM lets the wakeup radio adopt duty cycles and requires the data radio to keep sleeping always. A sender uses wakeup radio to send beacon packets continuously to activate its forwarder, and the forwarder, after successfully receiving the beacon packet, will turn on its data radio for the following data reception. Since each sender needs to spend an additional *setup latency* to wake up its forwarder, the performance of STEM reflects a typical trade-off between energy-efficiency and delay: the more energy-efficiency it wants to achieve, the higher latency it will have. A long setup latency also increases energy wastage since a sender needs to continuously transmit beacon messages before receiving replies, which impairs the energy gain from using duty cycles.

To address the tradeoff between energy-efficiency and end-to-end delay, Yang and Vaidya developed PTW [85] that employs a *pipelining* transmission technique to shorten the setup latency as well as achieve energy-efficiency. Similar to STEM, PTW uses dual radio setup, and one of them takes care of control message exchange by applying duty cycles. The pipelining is designed based on the observation that two

radios' transmission period can be overlapped, which saves the setup latency since a node can wakeup its forwarder while receiving data from another. However, its effectiveness largely depends on the data packet size and data radio bandwidth, meaning it still has considerable setup latencies when transmitting short data packets (which is usually true in WSN applications). To minimize duty cycles, PTW reduces the wakeup radio's active period by using tone to wake up forwarders, which saves more energy when no event happens. However, this mechanism also requires PTW to activate all the neighbors that receive a wakeup message since a receiver can not distinguish sender's address based on wakeup tone. This "large scale" wakeup causes a quick increase of energy consumption as more events happen in a dense WSN.

Dhanaraj *et al.* proposed LEEM [23] to further reduce setup latency without sacrificing energy-efficiency. LEEM also deploys two radios operating on different channels, and its basic idea is to reserve the channel before the real data transmission to save the setup latency. To support channel reservation, LEEM requires a system-wide synchronization, in which the wakeup times of the nodes along a route are scheduled sequentially. Through this reservation mechanism, LEEM eliminates intermediate nodes' setup latencies except the first hop.

STEM, PTW, and LEEM are all targeted toward event-driven WSN. However, they didn't address the spatially-correlated contentions prevalent in event-driven WSN and only provided results for the single source with light traffics. Sift [28], a MAC protocol proposed by Jamieson *et al.*, addressed the spatially-correlated contention by facilitating the first few reporting packets to reach the base station with a low latency. Sift uses a nonuniform probability distribution function to pick a transmission slot within the slotted contention window. Compared to 802.11 MAC protocl, Sift was shown to considerably decrease latency under spatially-correlated contentions. However, Sift does not address the energy-efficiency issues. It may increase the energy wastage caused by idle listening due to the requirement of listening to all slots before sending.

Zorzi *et al.* proposed a *Geographic Random Forwarding* (GeRaF) technique for general ad-hoc networks [95]. GeRaF assumes each node is aware of its location, and randomly selects forwarders based on the location information to improve energy-efficiency and delay. GeRaF also requires a two-radio setup to handle collisions, and it does not consider the special traffic pattern and spatially-correlated contention in WSN. Polastre *et al.* [58] develop a lightweight CSMA MAC protocol called B-MAC for sensor networks.

It adopts the LPL (low power listening) and clear channel sensing (CCA) technique to improve channel utilization and energy-efficiency. B-MAC provides a well-defined interface and can be easily reconfigured to meet different applications' requirements. Since B-MAC is deliberately designed to contain a small core of media access functionality, it does not provide multi-packet mechanisms like hidden terminal support or message fragmentation. Vuran and Akyildiz proposed a spatial correlation-based collaborative MAC (CC-MAC) that relieves contention and improves performance by reducing redundant nodes' reports [77]. However, since typical sensor applications usually require reports from multiple sources to improve reliability and reduce observed event distortion, spatially-correlated contention still exists in CC-MAC and can impair its performance.

TDMA MAC Protocols

TDMA protocols eliminate collisions and idle-listening overhead since each node's communication slot is pre-assigned [75, 76, 59], which makes them more energy-efficient under high traffic-load conditions. However, TDMA protocols are usually complicated since they require precise clock synchronization that incurs clock drift problems. Furthermore, TDMA protocols have a high control overhead at low traffic loads, since typically each node can only transmit data during its pre-assigned time slot.

TRAMA (for *TRaffic-Adaptive Medium Access*) [59] is a TDMA-based MAC protocol for WSN. It reduces energy consumption by eliminating collisions of packet transmissions and putting a node into a low-power sleep mode whenever it does not receive or transmit. TRAMA divides time into slots and uses a distributed election approach to determine nodes' occupancy of time slots based on traffic information. To utilize channel more efficiently, TRAMA allows nodes to reuse time slots if the original occupier does not want to transmit packets. By eliminating collisions and retransmissions, TRAMA achieves higher throughput and better energy-efficiency than most CSMA-based MAC protocols. However, it has higher latency and algorithmic complexity, which makes it "well suited for applications that are not delay sensitive but require high delivery guarantees and energy efficiency" [59].

LMAC (for *Lightweight Medium Access Control*) [75] is another TDMA-based MAC protocol for wireless sensor networks. LMAC organizes time into slots, and each active node is in control of one slot: a node attempting to send packets needs to wait until its time-slot comes, then send a header in the control slot followed by a data slot in which it can send data. LMAC ensures collision-free transmission by letting

each node select a slot number that is not used within a two-hop neighborhood. LMAC's drawback is that nodes must always listen to all control slots in a frame since other nodes may join the network at arbitrary moments, which leads to unnecessary energy wastage.

Hybrid MAC Protocols

Several MAC protocols have been proposed to combine the features of CSMA and TDMA protocols [42, 60, 88, 74] with the aim to incorporate the benefits from both of them. In these hybrid protocols, active/sleep duty cycles are applied by dividing time into frames during which a node spends a portion of the time for communication and sleep for the rest time to reduce the energy-wastage caused by idle listening. They inherit TDMA MAC protocols by using local synchronization to determine each node's communication period, and also simulate CSMA MAC protocols by allowing neighboring nodes to access channel in a CSMA manner within their active periods. Since the active period in each frame has to be long enough to contain one or more data packet transmissions, the duty cycles used in these hybrid protocols are usually higher than that in pure CSMA protocols, which limits their energy-savings.

S-MAC [88] is one of the first few protocols that combine CSMA and TDMA features. In S-MAC, each node adopts an active/sleep duty cycle to reduce the energy wastage caused by idle listening, and neighboring nodes form virtual clusters and share common sleep schedule through locally managed synchronization. S-MAC uses a CSMA-based manner, i.e., RTS/CTS control packets [7], to coordinate two nodes that attempt to access channel within the same active period. Since intermediate nodes in different clusters have their own sleep schedules, S-MAC sacrifice end-to-end latency for the sake of energy-efficiency. Furthermore, S-MAC requires periodically time synchronization (once per 10 seconds), which increases its protocol overhead.

The duty cycle used in S-MAC is fixed, which causes unnecessary energy wastage for handling low traffic and results in low throughput when sporadic events happen. To address this problem, Dam and Langendoen proposed T-MAC to improve S-MAC's energy-efficiency by introducing an adaptive duty cycle in which the listen period ends when no activation event has occurred for a time threshold. An activation event includes the firing of a periodic frame timer, the data reception, and the sensing of communication [74]. Since T-MAC breaks the synchronization of the listen periods within virtual clusters, it has the *early sleeping* problem in which a node goes to sleep when a neighbor still has data for it. T-MAC does not address the high delay issue in S-MAC; it still achieves energy-efficiency by sacrificing delay.

The tradeoff between energy-efficiency and delay has been explored intensively in recent years. DMAC [42] improves S-MAC and T-MAC in both end-to-end delay and energy-efficiency. DMAC was designed for the convergecast traffic pattern. Its key idea is a so-called *staggered* wakeup scheduling mechanism which schedules the nodes along a path to wakeup sequentially like a chain reaction. Staggered wakeup scheduling eliminates setup latencies except for the source node, so it enables DMAC to achieve a low latency but still be energy-efficient. DMAC also introduces additional contention handling techniques (e.g., data prediction and more-to-send (MTS) packet [42]) to improve network throughput under contention. However, DMAC constructs data-gathering trees to report events, which are not robust to node failures and prone to collisions since multiple nodes in the same level share a common forwarder. DMAC requires nodes in the same tree level to wake up at the same time, which can create intensive collisions when sources compete for close-by forwarders and impair throughput (i.e., the delay-throughput trade-off).

Rhee *et al.* proposed Z-MAC [60], which introduces a new idea of dynamically adjusting the MAC behavior between CSMA and TDMA depending on the level of contention in the network. Z-MAC uses loosely synchronization and topology information to determine the suitable MAC behavior: it behaves in a CSMA manner when contention is low, and in a TDMA manner in a high-contention network. Z-MAC's performance evaluation shows that it is well-suited to be applied in the applications with a medium/high contention. Recently, Wang *et al.* proposed HMAC (for *Hybrid* MAC) that attempts to combine the features of CSMA and TDMA MAC protocols [78]. To save energy wastage caused by idle listening, HMAC utilizes slotted frame structure to reduce a node's wakeup time period, and the frame structure is kept short to speed up packet transmissions. HMAC requires synchronization to assign slots for nodes, which introduces clock drift problems. Furthermore, HMAC does not guarantee collision-free medium access as typical TDMA protocols, so it still has the overhead of handling contention and collisions as CSMA MAC protocols. Pack *et al.* proposed TA-MAC (for *Task-Aware* MAC) [51], whose basic idea is to coordinate nodes' channel access probabilities based on their traffic load. The authors assumed each node's traffic load can be determined in advance, so each node calculates its total traffic load as well as all its neighbors', and access channel with the probability proportional to its traffic load and inversely proportional to the sum of its neighbors'. Simulation results reveal that the TA-MAC protocol exhibits less collisions and hence leads to more energy savings. Though coordinating nodes' channel accesses based on their traffic load is a reasonable choice for

reducing collisions, the assumption of determining traffic load in advance is not always true and may restrict its practicability. Further, TA-MAC only considers the energy wastage caused by traffics (i.e., collisions), while ignoring the idle listening problem, which can significantly restrict its energy saving.

To enable a better understanding on the existing WSN MAC protocols, in the following we provide a brief summary on their features and point out the issues deserve further exploration.

In general, most WSN MAC protocols take energy-efficiency as their primary design objective, and duty cycles have been widely used to conserve energy. However, energy-efficiency can introduce tradeoffs to other performance metrics, e.g., delay and throughput. Earlier MAC protocols usually sacrifice delay to achieve energy-efficiency [88, 74, 67]. In the recent years researchers have been focused on designing energy-saving MAC without sacrificing latency [42, 23]. However, they usually require network-wide synchronization and/or dual radio setup, which increases protocol overhead and hardware cost, and brings the trade-off between protocol simplicity and performance. A preferable MAC protocol should improve both energy-efficiency and delay while keeping the protocol as simple as possible to satisfy typical WSN requirements.

Contention should be carefully addressed in designing MAC protocols since packet collisions can significantly degrade network throughput. This issue becomes more challenging in a sensor application due to the existence of spatially-correlated contention, which happens in a sporadic manner and is typically more intensive than traditional contention. Most existing WSN MAC do not provide sufficient mechanisms to handle it since they do not consider the specific traffic pattern in sensor applications, i.e., convergecast. In particular, most WSN MAC protocols are proposed for all-to-all communication pattern ([88, 74, 67], etc.) in which traffic are randomly distributed and hence cause less contention. Several protocols consider the contention for convergecast [42], however, they follow a data-gathering tree to transmit packets, which is prone to contention in nature.

Though convergecast introduces higher requirements for handling contention, it also provides chances to improve MAC design and possibly leads to a more efficient MAC protocol than the one designed for all-to-all traffic. We are interested in designing energy-saving WSN MAC protocols to improve convergecast's performance while minimizing protocol's overhead (e.g., synchronization, dual radio setup, etc.).

2.2.2 *Energy Hole Problem*

The energy hole problem is a specific problem for WSN that can significantly impair energy consumption balance and network lifetime. It has been explored extensively in recent years, and a lot of approaches have been proposed to address it. Several approaches balance energy consumption by adjusting nodes' transmission ranges according to their positions. Li *et al.* developed an analytical model to help understand the different energy consumption rate in WSN [36], which verifies that nodes in inner rings suffer much faster energy consumption rates and thus have much shorter expected lifetime. Based on this model, they investigated the effectiveness of several approaches used to address the energy hole problem, including deployment assistance, traffic compression, and aggregation. Perillo *et al.* proposed a model to study the optimal transmission range distribution to maximize network lifetime [53], and revealed the upper bound on the lifetime of several typical scenarios (e.g., chain topology network). Olariu and Stojmenović also assumed an adjustable transmission range at each sensor node [50], and divided a WSN into a set of concentric coronas whose width equals the transmission range of the nodes in it, so each corona contributes one hop in a route of packet transmission. They provided an iterative process for varying the corona width (transmission range) to address the energy hole problem: the closer to the base station, the shorter the transmission range.

Clustering has also been considered an effective approach to address the energy hole problem and prolong network lifetime [69, 86]. These approaches form clusters with different sizes to balance the workload of cluster heads that forward packets. In particular, the cluster heads close to the base station, which naturally carry higher traffic load, form smaller clusters to reduce their workload within a cluster and hence balance energy consumption. However, most clustering approaches require periodical re-clustering to balance energy consumption, which increases control overhead and causes communication interruptions. The energy hole problem can also be addressed by managing node distribution. The general idea is to increase the node density around the base station to reduce the average traffic load of the nodes in the "energy hole". Li *et al.* proposed a non-uniform sensor distribution strategy in which the density decreases as the distance to the base station increases [41]. They also designed a strategy to move the base station around such that load balancing can be obtained [40]. Chang *et al.* proposed a distance-based scheme and a density-based scheme [17]. The distance-based scheme, similar to [50], adjusts nodes' transmission ranges. The density one partitions the WSN into equal-sized zones, and adjusts the density in each zone to balance energy

consumption.

The major problem of most existing approaches is that they assumed communication as the only energy consumption source, while ignoring the energy consumed by idle listening that can even dominate the overall energy consumption in event-driven sensor applications. As a result, these approaches are unlikely to coincide with most WSN MAC protocols that endeavored to reduce energy consumed by idle listening (e.g., by using duty cycles). We will consider the energy consumed by both traffic and idle listening for designing approaches to address energy hole problem, and make it applicable for most WSN MAC protocols that use duty cycles. Another issue that is usually ignored by most existing approaches is the trade-off to the network performance. For instance, some approaches that address the energy hole problem can change the route length for a packet transmission and influence the end-to-end delay [50, 53]. In our work, we explicitly take network performance into the design consideration, and our approach addresses the energy hole problem while maintaining network performance (e.g., delay and throughput).

2.3 ECR-MAC

2.3.1 Assumptions

We make the following assumptions about WSN, which motivate us to design ECR-MAC and make it suit to typical sensor applications. We assume WSN is typically over-provisioned, i.e., deploying a large number of nodes with a high density. We assume there is only one base station that collects data from sensor nodes, and assume sensor nodes are required to collaborate for sensing and reporting events, so shortening the delay of the first few reports and improving the network-wide performance (e.g., throughput) is more important than addressing the fairness among senders.

The major design objective of ECR-MAC is to improve performance for convergecast, which is usually the predominant traffic pattern and is critical to ensure the performance of typical sensor network applications [42, 31]. In WSN there may also exist other traffic patterns besides convergecast, e.g., queries from the base station to sensor nodes (i.e., *forward direction traffic* [31]). ECR-MAC can also improve these traffics' performance with the aid of location information. Since the methods to handle convergecast and forward direction traffic are similar, we first assume all traffics are convergecast and present ECR-MAC, then briefly describe how ECR-MAC handles forward direction traffic.

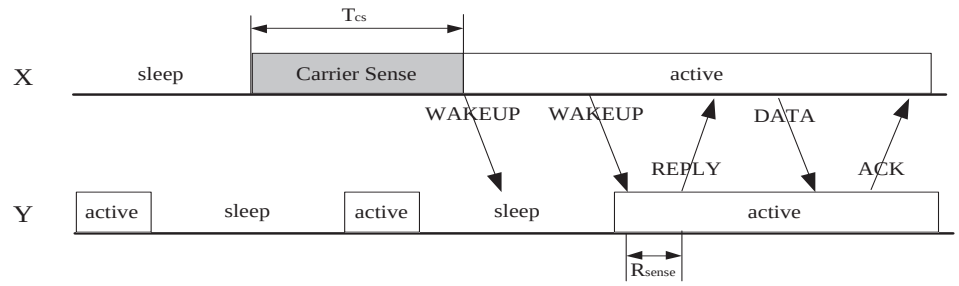


Figure 2.1: Basic Protocol Illustration

2.3.2 Basic Approach

The basic idea behind ECR-MAC is to add flexibility for packet forwarding, i.e., instead of waiting for a particular forwarder, each sender "hurls" packets as quickly as possible to any one of the nodes (termed *potential forwarder*) that can help transmit packets, which can effectively shorten the setup latency, reduce the transmission energy consumption, and eliminate the requirement of synchronization since in a dense WSN there usually exists sufficient potential forwarders that can serve for a sender. ECR-MAC uses active/sleep duty cycles and operates as follows: each node will be periodically activated for T_{active} to detect any packet-forwarding requirements. Before transmitting packets, a sender x will sense for a long enough duration T_{cs} to detect on-going communication. If none is detected, x sends WAKEUP messages periodically to see if one of its potential forwarders is awake. Any x's potential forwarder that receives a WAKEUP message will send a REPLY message after sensing for a short random time R_{sense}, which effectively reduces REPLY collisions from neighboring potential forwarders. Upon receiving the first REPLY from a potential forwarder y, x sends data to y. y then becomes x's real forwarder and replies an ACK message. We provide an illustration of the basic approach in Figure 2.1.

We use T_{wakeup} and T_{reply} to denote the transmission delay of WAKEUP and REPLY messages. The time interval between sending WAKEUP messages is the expected delay of a REPLY message and equals $R_{sense} + T_{reply}$ (and, typically $R_{sense} \ll T_{reply}$). Note that T_{active} should be at least $2 \times T_{wakeup} + R_{sense} + T_{reply}$ long to ensure an active potential forwarder to receive WAKEUP messages [67]. We shorten WAKEUP and REPLY's packet length to reduce T_{active} and minimize the duty cycle. However, carrier sense itself cannot eliminate REPLY collisions from non-neighboring forwarders. The possibility of these REPLY collisions can be ignored when duty cycles are low, but it increases as higher duty cycles are used.

An obvious method to handle REPLY collisions is to let a potential forwarder randomly back-off a period within a contention window $T_{cw} = kT_{reply}$ (k is an integer) before sending REPLY. However, it significantly increases T_{active} (T_{active} should cover T_{cw}) and hence impairs the energy-efficiency. In ECR-MAC, we handle this problem by letting the sender x, after receiving colliding messages, resend a WAKEUP message to notify the potential forwarders that cause collisions. Only at this time the corresponding potential forwarders randomly backoff a time period within T_{cw} to send a second REPLY, and also randomly adjust their wake up time slots to avoid future collisions. Since most sensor applications use a low duty cycle where REPLY collisions rarely happen, this method can effectively save energy for most sensor applications, and also ensures a sender to find a real forwarder when using higher duty cycles.

Since a sender's real forwarder is dynamically selected, we termed this mechanism *Dynamic Forwarder Selection* (DFS). Clearly, DFS exploits multiple forwarders to facilitate packet transmissions. Assume a sender has *K* potential forwarders, then ideally DFS can shorten a sender's setup latency from $\frac{T_{sleep}}{2}$ to $\frac{T_{sleep}}{2K}$ if these potential forwarders' wake up time slots are evenly spaced. To reduce computation and communication overhead, we let each node randomly choose wakeup time slots but with the same T_{sleep} and T_{active}, which eliminates the synchronization and is likely to evenly distribute potential forwarders' wakeup time slots (we will address wakeup times' even-distribution issue in the next chapter by designing topology control techniques). So by deploying DFS, ECR-MAC can achieve a short end-to-end delay even using a large T_{sleep}, and hence improve both delay and energy-efficiency. Since higher network density provides more potential forwarders for each node, ECR-MAC can scale well with the network density.

We design a simple but efficient flooding technique to assign potential forwarders for each node. A WSN is organized into a set of rings centered at the base station, and the level of the ring where a node is located corresponds to the number of hops of the shortest path from that node to the base station. The nodes in the inner rings (closer to the base station) are assigned to be potential forwarders of the nodes in the next outer ring. Each node's ring level can be efficiently identified with an initial set-up by flooding a message from the base station and identifying the hop distance. In practical scenarios different nodes may have varying number of potential forwarders due to irregular node distribution. To avoid nodes that have few potential forwarders to become bottlenecks for packet transmissions, we require a node that is in the same ring but has more potential forwarders than the sender (exceed a threshold) to also serve for packet

forwarding, which may increase the number of hops and nodes involved for packet transmission, but can greatly shorten T_{setup} of the nodes that have few potential forwarders.

2.3.3 Handle Contention

Spatially-correlated contention is common in typical sensor applications and should be efficiently handled to ensure protocol performance. Data aggregation is an effective technique to reduce traffic and relieve contention since it reduces the number of traffics in the network. ECR-MAC can also benefit from data aggregation when handling contention, but the discussion about data aggregation is beyond the scope of this dissertation and readers are referred to [49]. In the following we will present the contention-handling techniques used in ECR-MAC to improve throughput and minimize energy wastage.

Network Throughput

WSN nodes usually collaborate for sensing and reporting events, so instead of considering each source's independent throughput, we focus on the network throughput, i.e., the total number of reports for a certain event that can be received by the base station within a certain time period. Increasing the network throughput can effectively improve event-to-sink reliability [8].

Spatially-correlated contention is caused by simultaneous packet transmissions, so an effective contention-handling method is to let senders avoid competing for forwarders at the same time. In ECR-MAC, we deliberately avoid synchronization to not only reduce overhead, but also relieve contention, since certain type of synchronization, which let senders and/or forwarders wake up at the same time (as in [23, 42]), can intensify the contention caused by senders' competition for packet forwarding. In ECR-MAC, each node maintains multiple potential forwarders that choose their own wakeup schedule randomly and independently, so a sender, as soon as its report is ready to be forwarded, can usually identify a forwarder quickly without competing with other senders whose reports may be ready at a later time, which disperses the time for different senders' packet forwarding and improves network throughput. Even if two senders send reports at the exact same time, ECR-MAC can also relieve contention by dispersing the paths the senders will take. In DFS, the potential forwarder that first successfully receives WAKEUP will become the real forwarder, which allows the node in the network area with less contention to win the forwarder competition. So DFS relieves contention by allowing senders to deploy independent routes that detour the congested network area, and hence

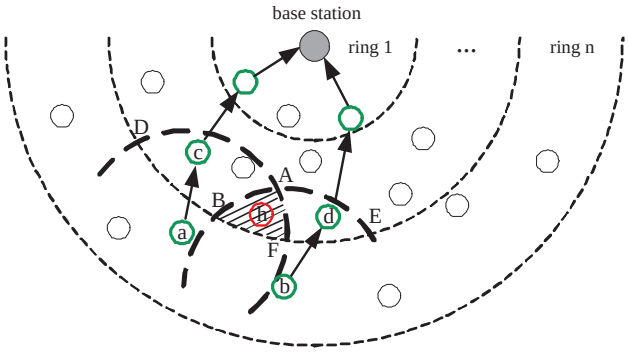


Figure 2.2: An Event Reporting in ECR-MAC

improve the network throughput. In Figure 2.2, we illustrate an ideal scenario for ECR-MAC: both nodes *a* and *b* sense a common event and report to the base station at almost the same time. They independently wake up their forwarders by performing the DFS procedure. Since the forwarders in the contention area (e.g., the node *h* in the shadow area) can not receive WAKEUP messages due to collisions, the real forwarders, *c* and *d*, will be located at the area without collisions, which relieves the contention by dispersing the routes in the following packet transmissions and improves network throughput.

Energy Wastage Caused by Contention

Network contention, though is effectively relieved by DFS, still exists in the area close to the base station that is the converging point of event-reports. It may cause overhearing and packet collisions, the two major sources of energy wastage that should be avoided. In ECR-MAC, we design several techniques to avoid overhearing and efficiently handle collisions. To avoid overhearing, a sender will sleep for a certain period of time T_s if it hears communication from neighbors during its carrier sense period, and the length of sleep time is estimated based on the duration that its neighbor spends for a packet transmission/forwarding. In typical sensor applications, fairness among senders is usually unimportant since all reports are for a common event [9], while it is useful to let the first few reports reach the base station as soon as possible to allow a quick response to the event [28]. So we further give the nodes at lower ring levels (closer to the base station) higher priority when transmitting packets since they are likely to reach the base station more quickly, and hence shorten the delay of the first few reports for an event. In particular, if a sender *A* overhears a message from another node *B* (can be either sender or receiver) with a higher priority, *A* will wait for additional period besides T_s to let *B* occupy the channel for the following packet transmissions.

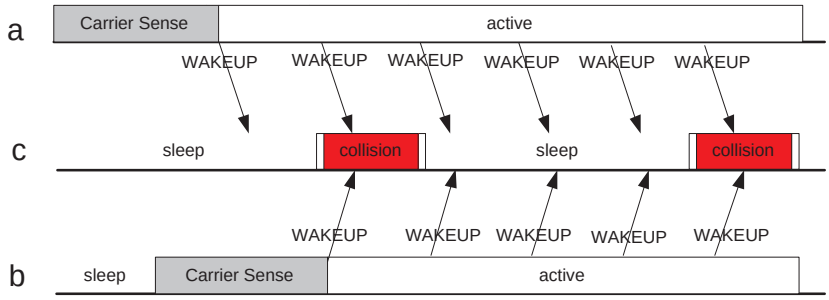


Figure 2.3: Control Collision Illustration

By avoiding overhearing, we essentially eliminate collisions from neighboring nodes. However, collisions still exist from non-neighboring nodes. Hidden terminal problem is a well-known problem that cause collisions among non-neighboring nodes. Traditional methods handle it by using virtual carrier sense and RTS/CTS mechanisms [7], however, they become less effective if duty cycles are applied, since each sender needs to send control messages (RTS or WAKEUP in ECR-MAC) periodically with a high frequency (to shorten T_{active}) to wake up a forwarder, which makes these control messages prone to collisions. We illustrate such a control collision problem in Figure 2.3: two non-neighboring nodes a and b detect a common event and report to the base station. They each periodically send WAKEUP messages to wake up their potential forwarders, however, their message transmission period is overlapped (which is very possible due to the high frequency for sending WAKEUP messages). So a forwarder c that can be reached by both a and b will keep receiving corrupted messages during their active periods. Without effective control, it can cause the worst result: two senders keep sending WAKEUP messages forever without receiving any replies, and we call such senders *dead senders*. In most MAC protocols that use duty cycles, each node has a unique forwarder, which makes these protocols prone to suffer control collision problems. A popular approach ([42, 23, 88, 74]) to address control collision is to use the back-off mechanism, i.e., let a node randomly back-off a time period before sending WAKEUP messages. However, it will significantly impair protocol's energy-efficiency due to the enlargement of T_{active} that should cover the back-off time. Another approach to handle this problem is to activate the data radios of all the neighbors that sense collision in their control radio, so the intended receiver can eventually receive the following data [67, 85]. This method, however, can cause large energy-wastage in a dense network since a lot of nodes that are not forwarders, but can be reached by both senders, will be activated unnecessarily.

ECR-MAC can handle the control collision problem efficiently: each node has multiple potential forwarders, it is unlikely that two non-neighboring senders can reach each other's all potential forwarders. As shown in Figure 2.2, the two sender a and b's potential forwarders will be located at the area encompassed by $\widehat{DAF}$ and $\widehat{EAB}$ respectively, which do not completely overlap each other except in the ring 1 where the base station is located (note that the collision area $\widehat{ABF}$ is usually a proper subset of $\widehat{DAF}$ and $\widehat{EAB}$). So the influence of control collision problems is minimized. To completely eliminate dead senders, we require a sender to randomly backoff a time if it does not get any REPLY within a $T_{active} + T_{sleep}$ period after sending the first WAKEUP (note it happens very infrequently). This way, ECR-MAC eliminates dead senders and avoids collisions with very low overhead.

2.3.4 Handling Forward Direction Traffics

ECR-MAC can also exploit DFS mechanism to improve the performance of forward direction traffics with the aid of location information. Localization in WSN is well-researched and there exists a lot of techniques that achieve high localization accuracy with low overhead, e.g., [27]. We assume each node knows its own location, and the sender and destination's locations are contained in the WAKEUP messages. Then each node a, after receiving a WAKEUP from a sender s, can determine if it should forward for s by comparing the distances $|ad|$ and $|sd|$ (d denotes the destination), thus similar DFS mechanism can be used for forward direction traffics. It should be noted that in the forward direction traffic the last hop transmission does not benefit from DFS since there is only one receiver (the destination) can receive the packet.

2.3.5 Analysis

By reducing setup latencies, ECR-MAC can effectively shorten packets' end-to-end delay even using a small duty cycle. However, it allows each node choose its wakeup time independently, which may cause REPLY collisions that affect delay when a high duty cycle is used. To provide a thorough understanding of ECR-MAC's effectiveness on shortening end-to-end delay, we analyze ECR-MAC's performance in terms of end-to-end delay for convergecast. To simplify the analysis, we assume all sensor nodes are uniformly distributed in a network area of density ρ nodes/m^2 with the base station at the center, and for each sender its potential forwarders' wakeup time slots are evenly distributed across a time period $T_{sleep} + T_{active}$. We assume an N-hop away source sends a packet to a base station, use $PF = \{p_i\}$ to denote a sender's potential

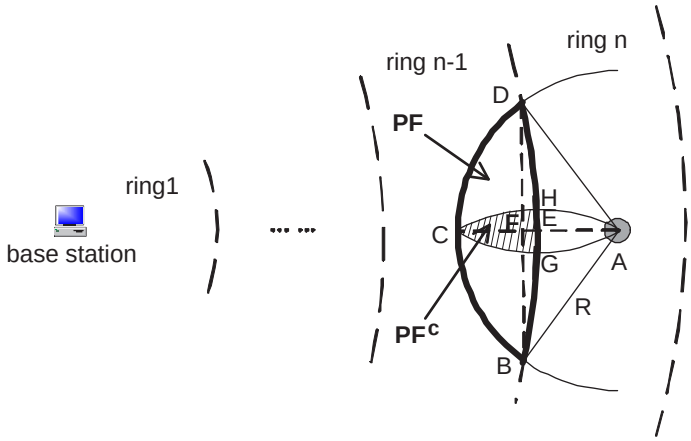


Figure 2.4: Illustration for REPLY Collision Analysis

forwarder set, $K = |PF|$ to denote the number of potential forwarders per node, R to represent nodes' radio range, and γ to denote the duty cycle.

We use d to denote the total end-to-end delay, and d' to denote the delay for each hop. We have:

$$d = d' \times N = (T_{setup} + T_{cs} + T_{tr} + T_{coll}) \times N \tag{2.1}$$

where T_{setup} denotes the setup latency, T_{cs} denotes the carrier sense delay, $T_{tr} = T_{wakeup} + R_{sense} + T_{reply} + T_{data} + T_{ack}$ denotes the transmission delay, and T_{coll} denotes the delay caused by REPLY collisions. For T_{setup} we have:

$$T_{setup} = \begin{cases} \frac{T_{sleep} - (K-1)T_{active}}{2K} & \text{if} \quad T_{sleep} \geq (K-1)T_{active} \\ 0 & \text{if} \quad T_{sleep} < (K-1)T_{active} \end{cases} \tag{2.2}$$

To derive T_{coll}, we show an illustration in Figure 2.4, where a sender in position A of the ring n is sending packets to the base station. On the average, A's potential forwarders should be 1/2 hops closer to the base station, i.e., they should be located at the area $\widehat{BCDE}$ (encompassed by thick curves) in which $\overline{CE} \approx \overline{CF} = \frac{\overline{AC}}{2} = \frac{R}{2}$. So the average number of potential forwarders can be estimated as:

$$K = \rho \times (\pi R^2/3 - \sqrt{3}R^2/4) \tag{2.3}$$

Some of these forwarders are common neighbors of all p_i, and they form a *Common Potential Forwarders* set $PF^c = \{p_i^c\} \subseteq PF$. In Figure 2.4, p_i^c should be located in the shaded area $\widehat{CGH}$, and we have:

$$\frac{|PF^c|}{K} = \frac{\pi R^2/6 - \sqrt{3}R^2/4}{\pi R^2/3 - \sqrt{3}R^2/4} \approx 0.15 \tag{2.4}$$

After *A* sends out WAKEUP messages, if a p_i^c wakes up first, there will be no REPLY collisions since all other potential forwarders can sense p_i^c's REPLY packet. So REPLY collisions only happens when two non-neighboring nodes *B* and *C* wake up first and $B \in (PF - PF^c), C \in (PF - PF^c)$. Thus we have the REPLY-collision probability:

$$Pr_{coll} = (1 - \frac{|PF^c|}{K})(1 - (1-\gamma)^\beta) = 0.85(1 - (1-\gamma)^\beta) \tag{2.5}$$

where β denotes the number of nodes in $PF - PF^c$ that are not the neighbor of the forwarder that wakes up first, and in a uniform network it can be estimated as $\beta = \frac{K - |PF^c|}{4} \approx 0.21K$. In ECR-MAC, each potential forwarder that detects a REPLY collision will randomly backoff a time period within T_{cw} before sending the second REPLY (in Section 2.3.2), so we assume on average that it takes $T_{cw}/2$ to handle a REPLY collision, and we have:

$$T_{coll} = Pr_{coll} \times (T_{wakeup} + T_{reply} + T_{cw}/2) \tag{2.6}$$

From equation 2.1,2.2, and 2.6, we can observe that d is in general dominated by T_{setup} and T_{coll} (T_{cs} and T_{tr} is fixed and smaller). When the duty cycle γ is low (which is assumed in typical WSN applications to achieve energy-efficiency), T_{setup} dominates d, so ECR-MAC effectively reduces d by almost a factor of K. As γ increases, d decreases until T_{coll} dominates it. Since T_{coll} is bounded by $0.85(T_{wakeup}+T_{reply}+T_{cw}/2)$ (when $\gamma = 1$), d will still be a small value even with a high duty cycle.

2.4 Addressing Energy Hole Problem

In the ECR-MAC proposed in the previous section, we let all nodes deploy the same duty cycle. Though using duty cycles effectively reduce energy wastage of idle listening, a uniform duty cycle assignment can cause the node closer to the base station to deplete energy more quickly than others due to the convergecast traffic pattern. To address this energy hole problem and hence improve network lifetime, we propose a

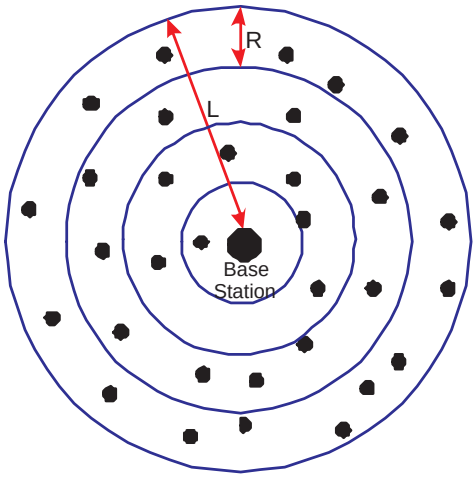


Figure 2.5: Concentric Corona Illustration

differential duty cycle assignment approach that balances the energy consumption of all the nodes in a network. Our differential approach can be applied in most WSN MAC protocols that utilize duty cycles besides ECR-MAC, as long as they are not critically dependent on all nodes using the same duty cycle, e.g., synchronized staggering wakeup mechanisms [42].

For the sake of analysis, we assume all sensor nodes are uniformly distributed within a disk of radius L from the base station. Since sensor nodes are typically inexpensive and simple in their hardware design, we do not assume the capability for a node to adjust its transmission range, i.e., all nodes have the same transmission range R all the time. Similar to [36, 50], we view the network as a set of *concentric coronas* whose width equals R, which is shown in Figure 2.5. So, roughly speaking, each corona contributes to one hop of a route from a source node to the base station.

We define network lifetime as the period till the first node depletes its energy [82, 69], and assume that all sensor nodes are static throughout the lifetime. We also assume that each node deploys a low duty cycle (as in ECR-MAC) to reduce energy wastage on idle listening, i.e., $T_{sleep} \gg T_{active}$. So T_{setup} dominates the hop delay, and it can be estimated as $T_{setup} = \frac{T_{sleep}}{2K}$, where K denotes the number of forwarders that each sender can have. Similar to [50], we assume an event can happen at any place in the network with the same probability, so each node has the same chance to detect events. Finally, as in [36], we assume the adopted MAC protocol can handle collisions efficiently (e.g., ECR-MAC), and ignore the energy wastage caused by collisions for the sake of analysis.

2.4.1 Workload of Nodes in Different Coronas

Several models have been proposed to analyze the energy consumption of the nodes at different levels or in different coronas [36, 50]. However, they ignored the energy consumption of idle listening, which can impair their applicability in realistic scenarios. To provide a more complete analysis, we will first assume all nodes deploy the same duty cycle γ, and analyze nodes' energy consumption due to both network traffic and idle listening. This analysis provides the foundation for designing differential duty cycle assignment next.

We number the corona (or level) as $0, 1, 2, \cdots, m$ starting from the base station, and the base station is the only one that is located in corona 0. For each node, its energy consumption is composed of three parts: E_l for the energy spent for idle listening; E_s for sending reports of detected events; and E_f for forwarding reports detected by others. Among these, E_s and E_f are caused by event-reporting traffic.

E_l is determined by the duty cycle γ, i.e., for each time unit we have:

$$E_l = \gamma e_{idle} \tag{2.7}$$

where e_{idle} denotes a node's energy consumption in idle listening state for a time unit. Clearly, E_l is the same for all the nodes if they have the same duty cycle γ.

E_s can be estimated based on the frequency at which events happen. Since each node has the same probability to detect an event, E_s is the same for all nodes in a network. Assume an event happens every T seconds, so a node has a probability of $P = \frac{k}{NT}$ to detect an event in each time unit, where N denotes the total number of nodes in a network, and k denotes the number of nodes that detect an event. Assume each node detecting an event will send q reports in a time unit, we have

$$E_s = P \times (t_t e_t) \times q = \frac{kq}{NT} \times (t_t e_t) \tag{2.8}$$

where t_t denotes the time a node spends for sending an event report, and e_t denotes a node's energy consumption in transmission for one time unit.

E_f is the part that distinguishes nodes' energy consumption based on its level. Since a node closer to the base station naturally needs to handle more traffic, it typically has a higher E_f than nodes further away

from the base station. We use S_i to denote the area of corona i. Since the width of a corona equals a node's transmission R, on average, a packet sent from corona i will travel through i nodes that are located in each of the corona i', $0 \leq i' < i$. So a node A in the corona i needs to forward packets for S_j/S_i nodes in the corona j, $j > i$, and we have:

$$E_f^i = P \times \frac{\sum_{i+1}^{m} S_j}{S_i}(t_t e_t + t_r e_r) \times q \tag{2.9}$$

where t_r denotes the time a node spends for receiving an event report, and e_r denotes a node's energy consumption in receiving state for one time unit.

We use E_i to denote the energy consumption per time unit of a node in corona i, and we have:

$$\begin{aligned} E_i &= E_l + E_s + E_f^i \\ &= \gamma e_{idle} + \frac{kq}{NT}(t_t e_t \frac{m^2-(i-1)^2}{2i-1} + t_r e_r \frac{m^2-i^2}{2i-1}) \end{aligned} \tag{2.10}$$

In a sensor network with uniform node distribution, γ, e_{idle}, e_t, e_r, q, N, k, and m are either constant parameters or can be estimated in advance. Assume each node is aware of its forwarders' wakeup time slots (through local message exchange as in [88]), then t_t and t_r are similar and only depend on packet size and transmission rate. So E_i is determined by corona level i and event happening interval T. We follow the parameters used in [23] and set $e_{idle} = 12.36mW$, $e_t = 14.88mW$, $e_r = 12.5mW$, $\gamma = 2\%$, $m = 10$, $N = 1200$, $k = 8$, $q = 5$, $t_r = t_t = 25mSec$, and show E_i against corona i with different event interval T in Figure 2.6. Clearly, nodes' energy consumption increases as their distances from the base station decreases (i.e., energy hole problem), and a smaller T, i.e., events happening more frequently, leads to a bigger energy consumption difference among different coronas (and hence deteriorate the energy hole problem), since it increases the weight of traffic energy consumption ($E_s + E_f^i$). Furthermore, we can observe from equation (2.10) that the energy hole problem will become more severe in a larger network (i.e., with a bigger m), since it increases the energy consumption differences among nodes in different coronas.

2.4.2 Balancing Energy Consumption

Nodes in different coronas have different traffic-related energy consumption (i.e., $E_s + E_f^i$ in equation (2.8) and (2.9)), so differentiating nodes' duty cycles, which essentially adjusts nodes' E_l (equation (2.7)), can compensate nodes' skewed energy consumption E_i and hence extend network lifetime. In addition,

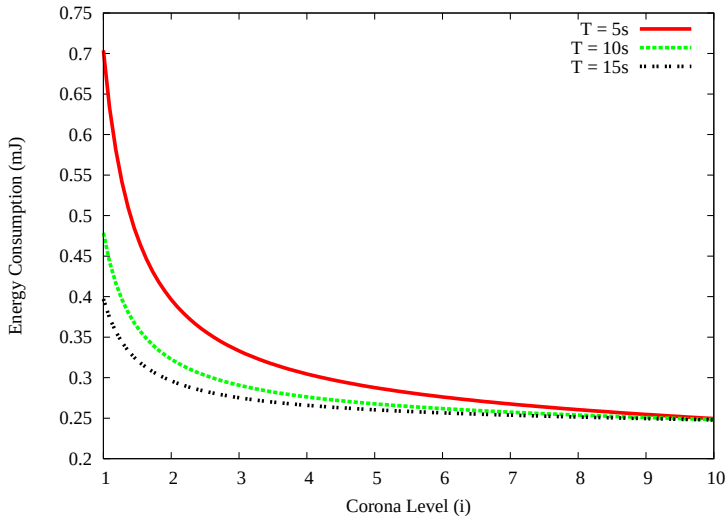


Figure 2.6: Energy Consumption Variation in Different Coronas

our approach that differentiates nodes' duty cycles (we call it *differential* approach hereafter) is designed to maintain network delay performance, i.e., it achieves lifetime extension without sacrificing end-to-end delay compared to the approach that assigns uniform duty cycles to all nodes (we call it *uniform* approach hereafter).

The duty cycle differentiation requires nodes in different coronas to deploy different duty cycles. Assume the duty cycle of a node in corona i is γ_i, hence its energy consumption of idle listening is $E_l^i = \gamma_i e_{idle}$. The energy consumption for sending and forwarding packets (E_s and E_f^i) is the same as in equation (2.8) and (2.9). We use E_i' to represent the energy consumption per time unit of a node in corona i in the differential approach, and we have:

$$\begin{aligned} E_i' &= E_l^i + E_s + E_f^i \\ &= \gamma_i e_{idle} + \frac{kq}{NT}(t_t e_t \frac{m^2-(i-1)^2}{2i-1} + t_r e_r \frac{m^2-i^2}{2i-1}) \end{aligned} \tag{2.11}$$

To balance nodes' energy consumption, we have:

$$E_i' = E_{i+1}', 1 \leq i < m \tag{2.12}$$

In the following we analyze the delay of a packet sent from the outermost (m) corona (has the highest end-to-end delay) to guarantee our differential approach achieve the same delay as that of the uniform approach.

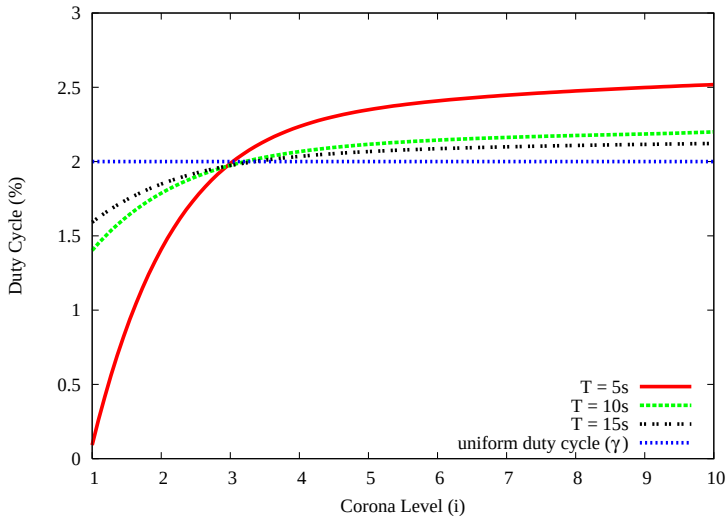


Figure 2.7: Duty Cycles in Different Coronas

Due to the deployment of duty cycles, in both approaches the hop delay is composed of the setup latency and the transmission delay. Note that typically a node's duty cycle is controlled by varying its sleep interval T_{sleep}, while a node's active period T_{active} is fixed to ensure receiving packets [67]. So a node that uses γ_i duty cycle will lead to a setup latency of $\frac{T_{sleep}}{2K} = \frac{T_{active}(1-\gamma_i)}{2K\gamma_i}$, where K denotes the average number of forwarders each sender can have. We use d_d to denote differential approach's end-to-end delay from corona m, and assume the base station is active all the time (i.e., $\gamma_0 = 1$). So we have:

$$\begin{aligned} d_d &= \sum_{i=1}^{m}(T_{setup}^{i} + t_t) \\ &= \sum_{i=1}^{m-1} \frac{T_{active}(1-\gamma_i)}{2K\gamma_i} + m \times t_t \end{aligned} \tag{2.13}$$

Uniform approach's end-to-end delay for a packet from corona m is $d_u = (m-1)\frac{T_{active}(1-\gamma)}{2K\gamma} + m \times t_t$. To ensure differential approach maintaining delay performance (i.e., $d_d = d_u$), we need:

$$\begin{aligned} \sum_{i=1}^{m-1} \frac{T_{active}(1-\gamma_i)}{2K\gamma_i} + mt_t &= (m-1)\frac{T_{active}(1-\gamma)}{2K\gamma} + mt_t \\ \Rightarrow \sum_{i=1}^{m-1} \frac{1-\gamma_i}{\gamma_i} &= (m-1)\frac{1-\gamma}{\gamma} \end{aligned} \tag{2.14}$$

Solving equation (2.11), (2.12), and (2.14) numerically provides the required γ_i for each corona. Since e_{idle}, e_t, e_r, q, N, k, m, t_t, and t_r are all constant parameters or can be estimated in advance, γ_i is determined

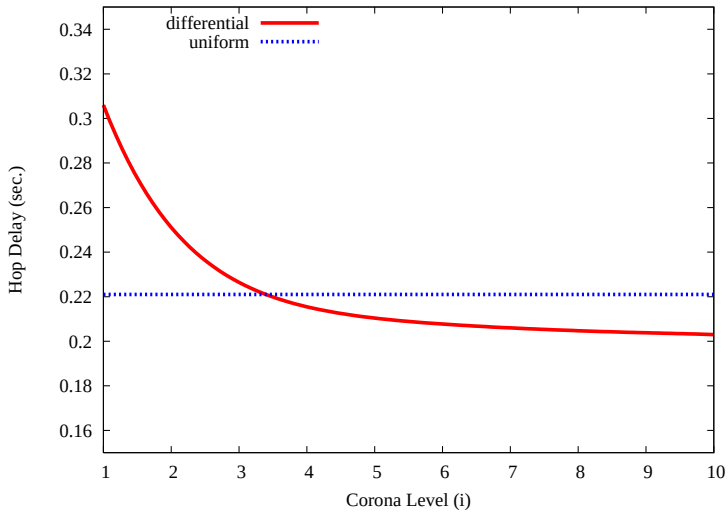


Figure 2.8: Hop Delay in Differential and Uniform Approach

by i, γ, and T. In Figure 2.7, we plotted the γ_i with different event interval T by following the same parameter setup as before ($\gamma = 2\%$). Since the duty cycles γ_i are used to compensate the un-balanced energy consumption in each corona, the duty cycle curve in Figure 2.7 shows an opposite trend as the energy consumption curve in Figure 2.6, i.e., the closer to the base station, the lower the duty cycle. Further, a smaller T, which represents a higher event frequency, increases the duty cycle difference among coronas since it enlarges the difference of traffic energy consumption (i.e., $E_s + E_f^i$) in different coronas.

In the differential approach, different duty cycle γ_i lead to varying hop delays in different coronas. In Figure 2.8 we show the expected hop delay (in $sec.$) in each corona for both uniform and differential approach. Clearly, in the differential approach the hop delay increases as a packet approaches the base station due to the decrease of γ_i, while uniform approach that has identical hop delay in each corona (equals $\frac{T_{active}(1-\gamma)}{2K\gamma} + t_t = 0.221sec$). As guaranteed in equation (2.14), differential approach has the same end-to-end delay as the uniform approach. So our differential approach can balance nodes' energy consumption and extend network lifetime (analyzed in the following) without sacrificing delay performance.

Network lifetime is dependent on energy bottlenecks. In the differential approach, all nodes should have similar energy consumption E_i', thus avoiding bottlenecks. While in the uniform approach, nodes in the corona 1 have the highest E_i due to the energy hole problem. So the lifetime extension ratio by using the

differential approach is:

$$\frac{\frac{1}{E_i'} - \frac{1}{E_1}}{\frac{1}{E_1}} = \frac{\gamma e_{idle} - \gamma_1 e_{idle}}{\gamma_1 e_{idle} + \frac{kq}{NT}(t_t e_t m^2 + t_r e_r (m^2 - 1))} \tag{2.15}$$

The denominator of equation (2.15) contains two parts: (1) $\gamma_1 e_{idle}$ which represents the energy consumed by idle listening for a node in the corona 1, i.e., E_l^1, and (2) the rest which represents the energy consumed by traffic for a node in corona 1, i.e., $E_s + E_f^1$. Assume e_{idle}, e_t, e_r, q, N, k, m, t_t, and t_r are all constant parameters or can be estimated in advance. Since γ_1 is determined by T and γ (as in Figure 2.7), the lifetime extension ratio depends on T and γ. For a given γ, when T is large, i.e., events rarely happen, E_l^1 dominates a node's energy consumption, so the lifetime extension increases as T decreases due to the decrease of E_l^1 (as shown in Figure 2.7, decreasing T leads to a smaller γ_1). However, when T is small, $E_s + E_f^1$ dominates a node's energy consumption, which makes the lifetime extension decreases due to the increase of $E_s + E_f^1$. So for a given sensor application and MAC protocol using duty cycles (with a given γ), the lifetime extension is dependent on the relationship between E_l^1 and $E_s + E_f^1$, which in turn depends on T, and there exists a certain event interval T_p that leads to the peak lifetime extension. Furthermore, varying γ values affect T_p, in particular, a smaller γ reduces the weight of E_l^1 and hence leads to a larger T_p. In Figure 2.9, we provide the analytical lifetime improvement ratio by equation (2.15) for different T and γ (note that we use $1/T$ in the x axis for clarity).

2.5 Performance Evaluation

We will conduct our performance evaluation in two steps. In the first step, we evaluate ECR-MAC's performance and compare it to several representative WSN MAC protocols. In the second step, we incorporated our differential duty cycle assignment approach into ECR-MAC, and focus on the performance improvement (e.g., lifetime extension, energy balance, delay, etc) over the original ECR-MAC with uniform duty cycles.

2.5.1 ECR-MAC

We have implemented ECR-MAC, using ns2 simulator [5], and compared it with four other CSMA MAC protocols that also exploit active/sleep duty cycles: STEM [67], PTW [85], LEEM [23], and DMAC [42].

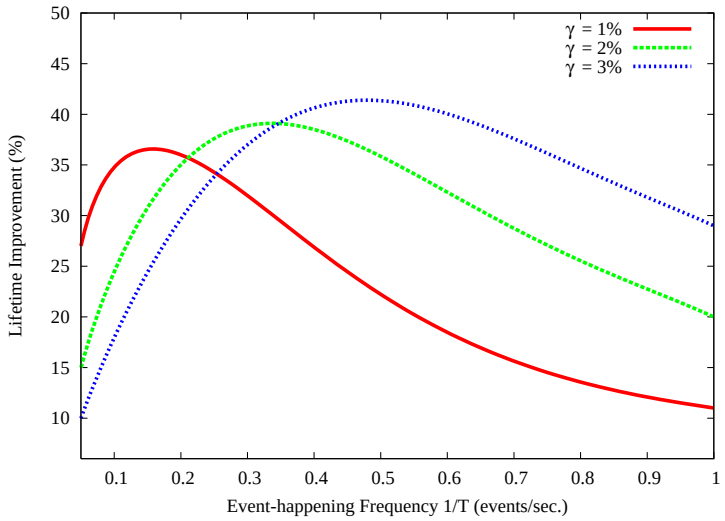


Figure 2.9: Analytical Lifetime Extension using the Differential Approach

Note that these four MAC protocols defer in terms of radio setup (STEM, PTW, and LEEM require dual radios, while DMAC uses one radio), wakeup scheduling (STEM uses *random* scheduling, PTW uses *pipelining* scheduling, LEEM and DMAC use similar *staggered* scheduling), and synchronization requirement (STEM and PTW do not require synchronization, while LEEM and DMAC require network-wide synchronization). So comparisons to these MAC enable us to evaluate ECR-MAC's performance thoroughly. Since these MAC protocols adopted two different bandwidth and energy consumption parameters in their original reports (the general parameters used in STEM, PTW, and LEEM are listed in Table 2.1, DMAC's are listed in Table 2.2), we conduct ECR-MAC's experiments for both parameter sets, which helps to provide an integrated evaluation for ECR-MAC in different sensor applications with different network conditions. For ease of description, we call the parameters in Table 2.1 as *low* parameters (they have a lower bandwidth), and the one in Table 2.2 as *high* parameters. For high parameters, we also include a fully-active MAC (IEEE 802.11) that does not use duty cycles for baseline comparison.

For both parameter sets, we conducted simulations in two scenarios: a simple single-source scenario, in which we compare ECR-MAC's basic performance to these four protocols, and a multi-source scenario to identify the protocols' performance in the realistic WSN with spatially-correlated contention. For both scenarios we randomly distribute 500 nodes in a $80m \times 80m$ network area. Each node's transmission range is set to 15m, and the data packet size is 64 bytes. We set ECR-MAC's T_{active} to be 88ms/2.2ms for the

Table 2.1: General Parameters (Low)

Parameter	Value
bandwidth (Kbps)	2.4
Transmit power (mW)	14.88
Receive power (mW)	12.50
Idle power (mW)	12.36
Sleep power (mW)	0.016

Table 2.2: General Parameters (High)

Parameter	Value
bandwidth (Kbps)	100
Transmit power (mW)	660
Receive power (mW)	395
Idle power (mW)	350
Sleep power (mW)	0

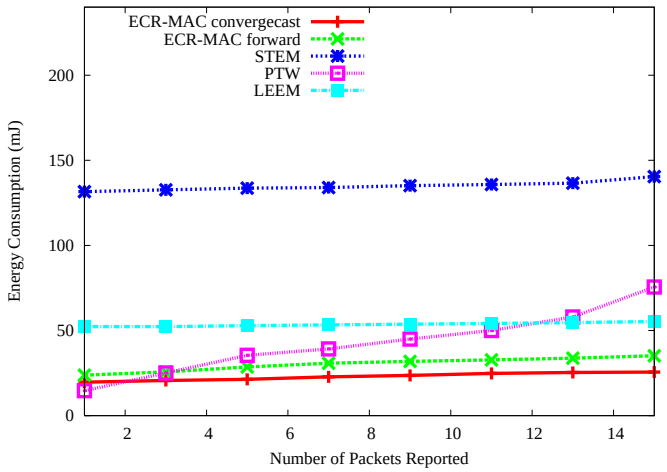
low/high parameters, respectively, which are sufficiently long to cover $2T_{wakeup} + R_{sense} + T_{reply}$. For both parameter sets we set ECR-MAC's duty cycle to be 1% except when we evaluate its performance with differential duty cycles. The protocol-specific parameters for STEM, PTW, LEEM, and DMAC are directly taken from [67], [85], [23], and [42], whose duty cycles are set to be 7.5%, 0.33%, 2.59%, and 10%, respectively. Similar to [23], we use flood routing protocol to set up routes for STEM, PTW, LEEM, and DMAC. For all these simulations, we assume a single base station located at the left bottom corner and remains active all the time. Each data point shown is the average of 20 independent simulations with different seed numbers. We run each simulation for 150/30 seconds with low/high parameters, sufficiently long for a typical event-reporting activity.

Performance of Single-source Scenario

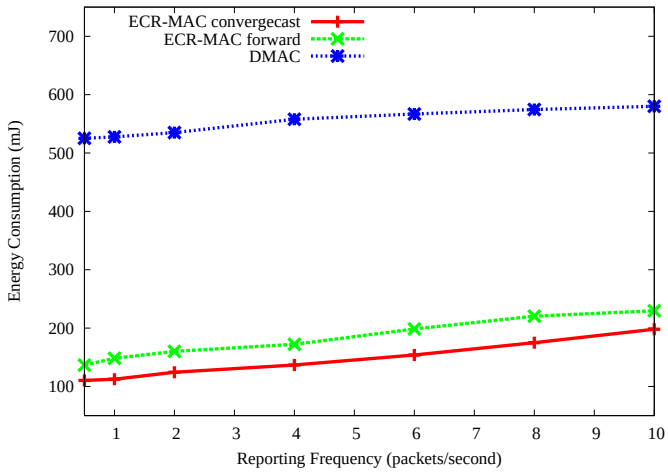
We provide simulation results in the single-source scenario to compare ECR-MAC's basic performance to STEM, PTW, LEEM, and DMAC. Similar to [23], we randomly select a source node (or a destination for forward direction traffic) that is 6 hops away from the base station to send/receive packets. In Figure 2.10 we show the energy consumption, averaged over all the nodes, against the number of packets reported. Clearly for both convergecast and forward direction traffic ECR-MAC consumes the least energy among

all the protocols, since it has a low duty cycle that minimizes idle listening, and its DFS mechanism can shorten the setup latency and hence save energy for transmitting packets. The forward direction traffic takes slightly higher energy consumption than convergecast due to its longer end-to-end delay (can be observed in Figure 2.11). In Figure 2.11, we provide the end-to-end delay against the number of hops. It should be noted that we show the ideal shortest hops in the x axis, which can be shorter than the hops of the real route identified by the routing protocol. The results of STEM, PTW, LEEM, and DMAC are close to those reported in [67], [85], [23], and [42]. ECR-MAC's convergecast delay is significantly less than STEM, but slightly higher than that of PTW and LEEM, because both PTW and LEEM adopt two radios and apply the pipeline technique to facilitate packet transmissions, while ECR-MAC, besides employing only one radio that prevents using pipeline technique, needs to afford the overhead to handle contention (e.g., WAKEUP/REPLY messages, carrier sense, etc), which increases its delay. Compared to DMAC, ECR-MAC has smaller delay for shorter paths, but its delay increases more quickly than DMAC's since DMAC utilizes network-wide synchronization to eliminate setup latencies in intermediate nodes along a route. The fully-active 802.11, which does not use duty cycles and hence does not have setup latency, has the smallest end-to-end delay. ECR-MAC's delay of forward direction traffic is significantly lower than STEM, but slightly higher than that of convergecast, since in the last hop ECR-MAC can not provide potential forwarders for forward direction traffic, while in convergecast the base station is always active.

To evaluate how duty cycles affect ECR-MAC's performance, we let a node 6 hops away from the base station send/receive a packet to/from the base station, and provide ECR-MAC's delay results against different duty cycles in Figure 2.12 (using low parameters). To reflect the effect of the number of potential forwarders K, we also provide the analytical results for $K = 2$ (as in [31]) for comparison. In this experiment setup we have $K = 11$ in ECR-MAC according to the formula (2.3). We can observe that the delay decreases quickly as the duty cycle increases from 1% to 10% (note it is plotted with a log scale for clarity), and slightly increases as the duty cycle exceeds 10% due to the increasing T_{coll}. In general, using the duty cycles between 2% to 5% can lead to the minimal delay while also achieving energy-efficiency. Analytical results are slightly lower than simulation results since we assumed an evenly-distributed wakeup time slots that is not necessarily satisfied in simulations. We can observe that increasing the number of potential forwarders, e.g., from 2 to 11, can significantly reduce delay when deploying a low duty cycle due to the



(a) Low Parameters



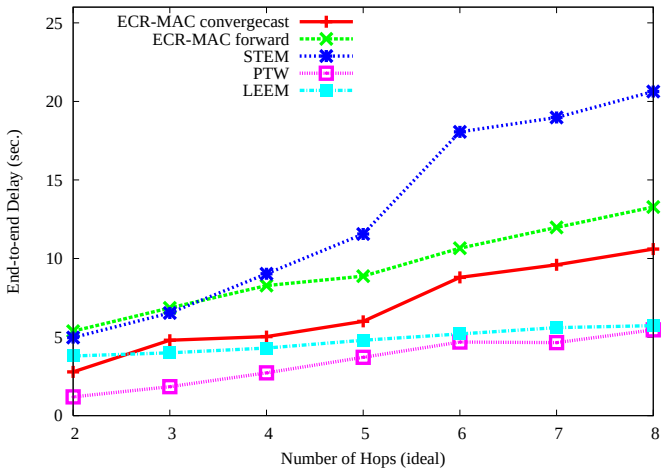
(b) High Parameters

Figure 2.10: Energy Consumption in Single-source Scenario

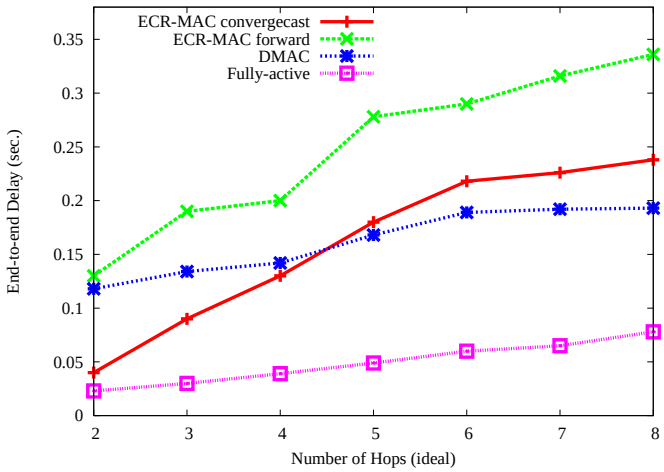
shortening of T_{setup}. Forward direction traffic has slightly higher delay than convergecast since DFS has no effect on the last hop of forward direction traffic.

Performance of Multi-source Scenario with Spatially-Correlated Contention

To evaluate protocol performance with spatially-correlated contention, we simulate a realistic scenario in which an event happens at a randomly selected place that is 6 hops away from the base station, and we assume it will be observed by the sensor nodes that are within 10 meters to this place. We do not perform evaluation for forward direction traffic for these experiments since spatially-correlated contention is specific



(a) Low Parameters



(b) High Parameters

Figure 2.11: End-to-end Delay in Single-source Scenario

for convergecast.

A typical event-reporting activity in realistic sensor applications is that multiple source nodes observing the same event send reports periodically. To reduce information redundancy but still achieve reasonable reliability of event estimation, it is desirable to let a subset of the nodes that detect a common event send reports [77], so we randomly select 8 sources from all the sources that detect the event, and show the protocol performance against the number of reports sent per source (by varying the reporting interval). In Figure 2.13, we provide the number of reports received by the base station within the simulation period, which essentially reflects the network throughput. ECR-MAC has higher throughput than other MAC protocol

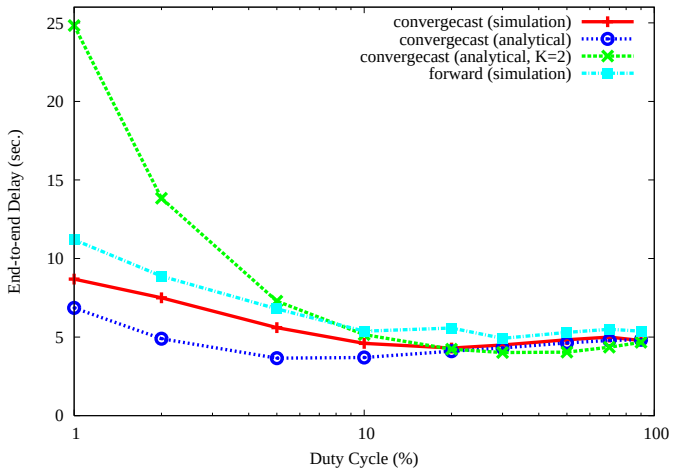
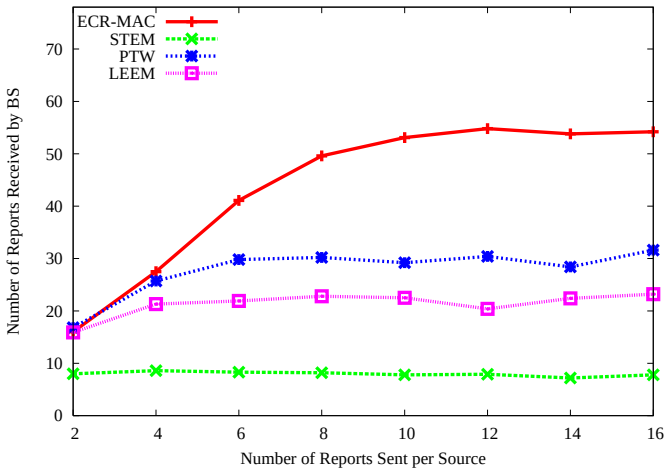


Figure 2.12: ECR-MAC's Performance against Different Duty Cycles

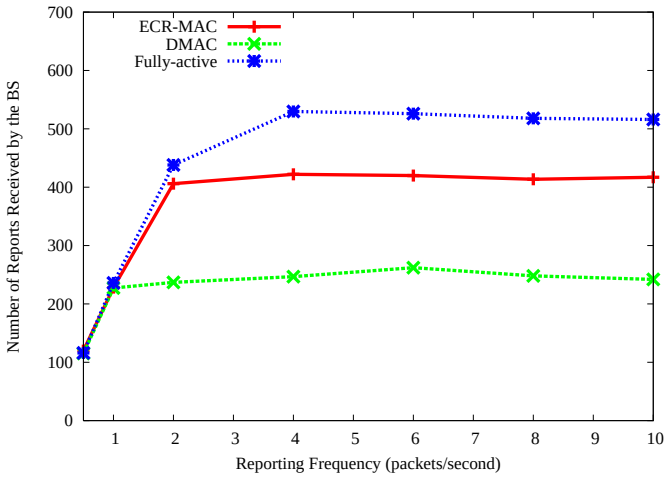
using duty cycles, since it handles contention by avoiding the use of synchronization and letting each sender take independent routes to detour the congested area. The fully-active MAC, which does not use duty cycles to save energy, naturally has the highest throughput. The other four protocols follow a convergence tree to transmit reports, which makes them prone to collisions and impairs their throughput. LEEM's throughput is lower than PTW, since it requires system-wide synchronization that activates forwarders at the same time, which brings more packet collisions when senders compete for forwarders. STEM has the lowest throughput due to its large setup latency.

In Figure 2.14, we show the energy consumption averaged over all nodes. Since ECR-MAC reduces the energy-wastage under contention, its energy consumption is the lowest among all the protocols, and it increases slowly as the increase of sources since more reports will travel through the network and reach the base station, while LEEM and STEM's energy consumption remain stable because their channel have been saturated even when each source sends very few reports (can be observed in Figure 2.13(a)). PTW's energy consumption increases quickly as more reports reach the base station since it handles collisions by activating all nodes detecting them and can cause large energy-wastage. We skip the energy consumption of fully-active MAC since its always-on radio leads to over 50 times higher energy consumption than ECR-MAC.

For some sensor applications, it is desirable to allow the first few reports to reach the base station as



(a) Low Parameters

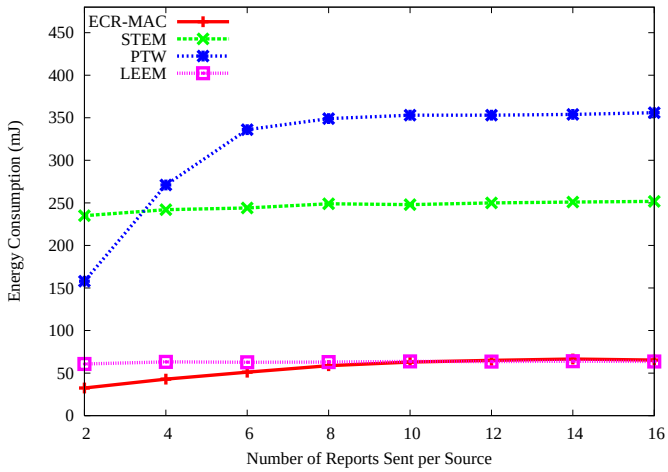


(b) High Parameters

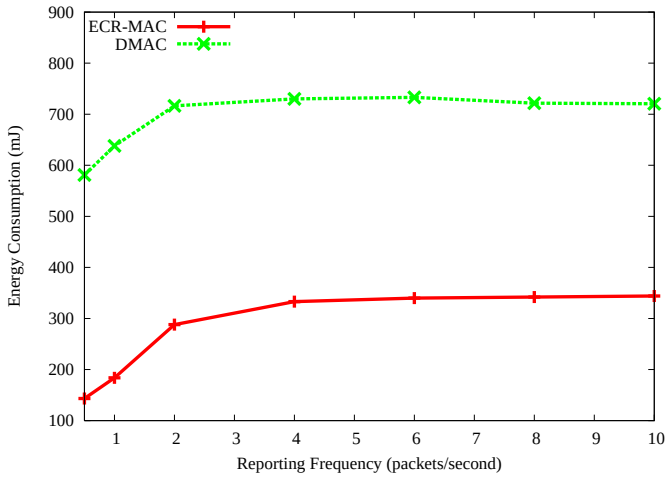
Figure 2.13: Network Throughput in Multi-source Scenario

soon as possible [28], which enables the base station to handle the event quickly. So we show the end-to-end delay of the first 10% reports in Figure 2.15. ECR-MAC, as expected, allows the first few reports to reach the base station with the lowest delay among all MAC protocols that use duty cycles, since besides deploying DFS mechanism that effectively relieves contention, it gives higher priority for the forwarders that are close to the base station to transmit packets.

To evaluate the scalability of ECR-MAC, we show protocols' performance against network density in Figure 2.16 and 2.17 by varying the number of nodes deployed in the network. We randomly selected 8 source nodes that detect the event, and let each source send 5 reports at a 30-second interval (we use low



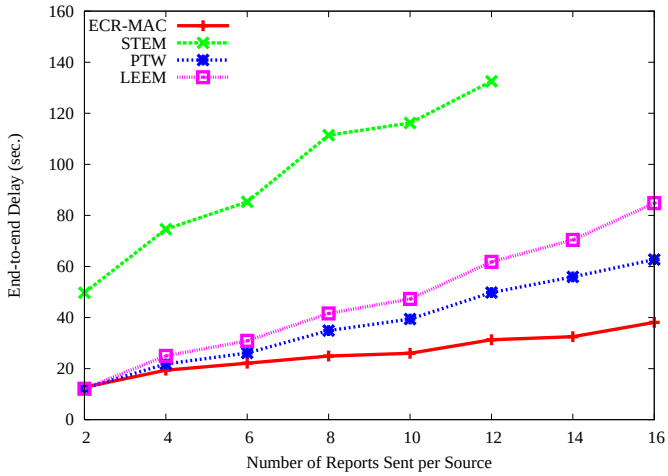
(a) Low Parameters



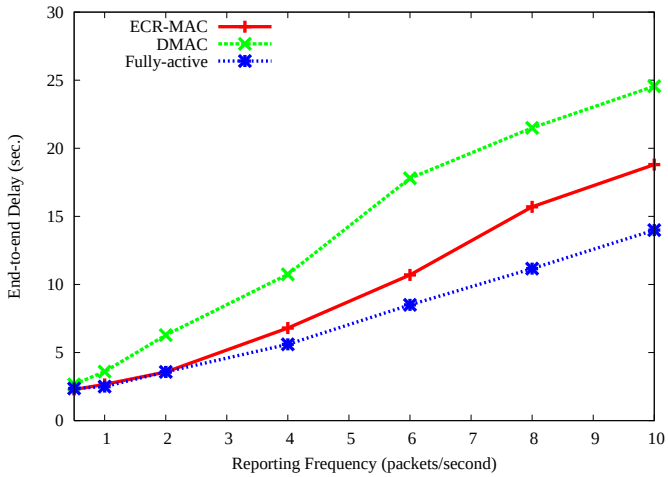
(b) High Parameters

Figure 2.14: Energy Consumption in Multi-source Scenario

parameters). Similar to [23], we provide the total instead of average energy consumption against network density in Figure 2.16: though ECR-MAC naturally consumes more energy as the network becomes denser, it has the slowest increase rate among all the protocols due to its efficient contention-handling mechanisms. In a sparse network (less than 20 neighbors per node), ECR-MAC's throughput is lower than PTW due to the small number of potential forwarders available for each sender, as shown in Figure 2.17. However, ECR-MAC's throughput increases quickly as the network becomes denser, since a denser network enables a sender to have more potential forwarders, which reduces the setup latency and facilitates transmissions. So ECR-MAC has a good scalability on the network density.



(a) Low Parameters



(b) High Parameters

Figure 2.15: End-to-end Delay of First 10% Reports

2.5.2 Test-bed Experimentation

To further evaluate ECR-MAC's performance on a realistic scenario, we have implemented ECR-MAC on MICA2 motes testbed [4]. In Table 2.3 we list the basic radio features of MICA2 that are obtained from [4]. For comparison, we also implemented a STEM-like protocol on motes, which uses the same random wakeup scheduling as in STEM but only uses one radio (MICA2 motes only support one radio), so this STEM-like protocol is supposed to achieve comparable performance as STEM (hereafter we call it STEM for ease of description). Finally, we also implemented a fully-active MAC protocol for baseline comparison, in which all nodes turn on their radios all the time.

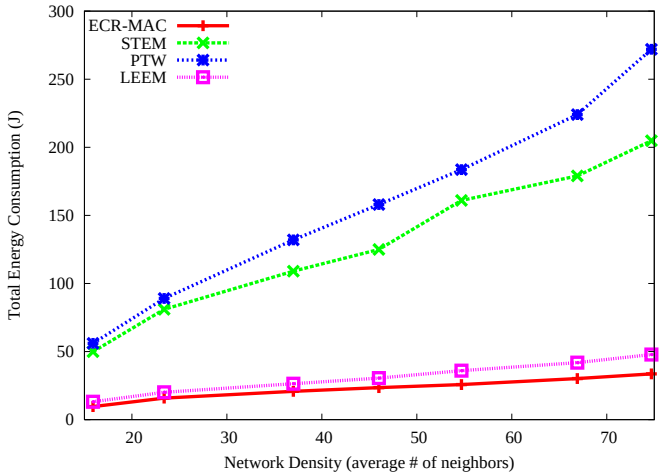


Figure 2.16: Energy Consumption against Network Density

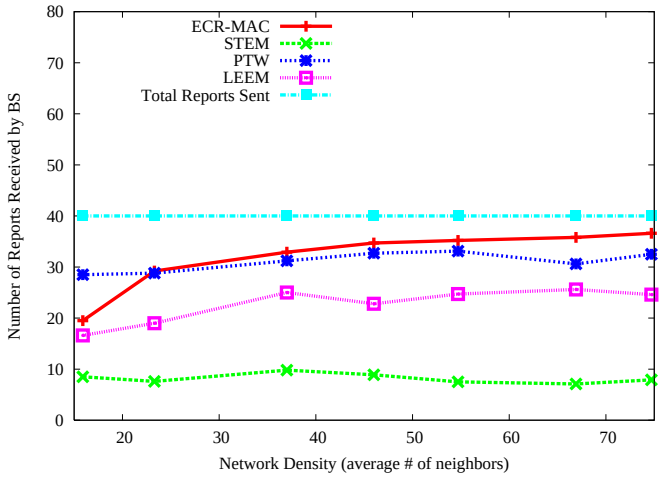


Figure 2.17: Network Throughput against Network Density

Table 2.3: Parameters of MICA2 Radio

Parameter	Value
Radio bandwidth (Kbps)	38.4
Data Packet size (bytes)	32
Transmit power (mA)	27
Receive power (mA)	10
Idle power (mA)	10
Sleep power (μA)	$<$1

Implementation

The default MAC used in motes is B-MAC [58]. B-MAC uses clear channel assessment (CCA) and packet backoffs for channel access management, low power listening (LPL) for energy conservation, and link layer acknowledgements for reliability. However, B-MAC is designed to contain a small and common core of media access functionality for general sensor applications, and it relies on higher layer services/protocols to address more sophisticated problems, e.g., the hidden terminal problem. So we implemented ECR-MAC, STEM, and a fully-active MAC on top of B-MAC to make full use of its efficient medium access mechanisms (e.g., CCA and LPL) as well as provide more sophisticated channel access support (e.g., addressing the hidden terminal problem). A side-effect of this implementation is the possible function duplicates, i.e., implementing similar functions more than once. For instance, B-MAC has provided the link layer acknowledgement mechanism, while both ECR-MAC and STEM also include such an acknowledgement. To minimize this redundancy, we choose broadcast as the common communication mechanism in B-MAC and set a destination address field in the packet to specify a particular receiver node for unicasts, which avoids redundant acknowledgements and simulates the omnidirectional message propagation in wireless networks. However, due to the lack of implementation details of B-MAC, there can still exist certain redundancies in implementation. For instance, the LPL mechanism used in B-MAC minimizes the idle listening period by using active/sleep duty cycles, which may cause duplicate effects as ECR-MAC and STEM also deploy duty cycles. These possible redundancies can cause additional power consumption and impair performance. However, since we implemented all ECR-MAC, STEM, and fully-active MAC on top of B-MAC, their performance results still provide a fair comparison among them.

An important difference between ns2 simulation and mote testbed experiments is the symmetry of communication links. Due to motes' different radio characteristics, there usually exist a considerable number of asymmetric links in the mote testbed, and this phenomenon becomes more severe if we reduce motes' transmission power (in the experiments we minimize motes' transmission power to facilitate network topology management). Asymmetric links, without careful addressing, can impair the effectiveness of dynamic forward selection mechanism in ECR-MAC since each node may identify a lot of useless forwarders that can not be reached when forwarding packets. To address this problem, we design a three-step handshake technique to replace the flooding technique used in the original ECR-MAC (and STEM) for identifying

potential forwarders. The three-step handshake includes three messages: route discovery, route reply, and route confirmation. A node *A*, after receiving a route discovery message from a node *B* that has fewer hops than it, will reply a route reply message. *B* will send back a route confirmation once receiving a route reply, and *A* will set *B* as its potential forwarder only if it receives a route confirmation from *B*. This three-step handshake effectively reduces the asymmetric links in the testbed (though can not eliminate them due to the unreliable wireless channel).

The implementation of ECR-MAC, STEM, and fully-active MAC use three major components provided by tinyos: *GenericComm*, *TimerC*, and *LogicalTime*. GenericComm provides packet sending and receiving interfaces that support basic communication functions. It also controls the radio state, i.e., turning on/off radios that are necessary for duty cycle implementations. TimerC is the component that provides timer functions. In ECR-MAC, STEM, and fully-active MAC, we need to set up several timers to implement the duty cycles, sensing period, back-off period, and packet retransmissions for handling packet losses. TimerC provides timers with a 1 ms granularity and hence effectively supports them. LogicalTime provides the interfaces to obtain local time for each node, and can be used to measure end-to-end delay in ECR-MAC. However, since different nodes' local times are not synchronized, we implemented a simple synchronization technique (similar to [56]) to ensure all nodes to have the same local time as the base station, which enables us to measure the end-to-end delay (note that ECR-MAC itself does not require synchronization).

Experiments

Our test-bed experiments involve 16 motes that form a topology illustrated in Figure 2.18, in which the farthest nodes are 5 hops away from the base station. A link between two nodes represents that they are within each other's transmission range. We set the MICA2 radio to transmit using the smallest transmission power (output power -20dBm) so that a node's transmission range is about 0.6m [42]. It should be noted that during experiments we observed different nodes' transmission range is not exactly the same and a single node's transmission range is not stable over time due to possible interfaces and battery depletion, so the topology illustrated in Figure 2.18 is only plotted in the general sense. In ECR-MAC, we set $T_{active} = 130ms$, and $T_{sleep} = 6.5s$, so the corresponding duty cycle is about 2%. As observed from Figure 2.18, each node can identify 2-3 potential forwarders in ECR-MAC. We set STEM's $T_{active} = 225ms$ and duty cycle to be 7.5% as in [67].

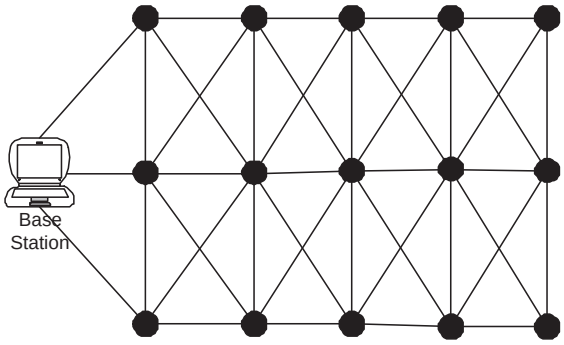


Figure 2.18: Test-bed Topology Illustration

Similar to our ns-2 simulations, we conducted test-bed experiments in two different scenarios: single-source scenario and multi-source scenario. For both scenarios we set the experiment time to be 50 seconds, sufficiently long for an event reporting activity. In Figure 2.19 we provide the end-to-end delay against the number of hops. Clearly, ECR-MAC provides a significant shorter end-to-end delay than STEM due to its dynamic forwarder selection mechanism. So even in a WSN in which each node can only identify a small number of potential forwarders (less than 3), the dynamic forwarder selection mechanism still effectively reduces setup latencies. The fully-active MAC has the lowest end-to-end delay since it does not have setup latencies. In Figure 2.20, we show the radio energy consumption of ECR-MAC and STEM against the reporting frequency. We can observe that ECR-MAC, which deploys a lower duty cycle and effectively reduces the setup latency, consumes less energy than STEM. However, since each node can identified a limited number of potential forwarders in the small-scale test-bed, ECR-MAC needs to spend more energy to wake up its forwarders when a low duty cycle is used, which restricts its energy-efficiency improvement compared to our simulation results where a large scale network with dense node deployment is assumed. We do not include the fully-active MAC's energy consumption since it is about 15 times higher than ECR-MAC.

In Figure 2.21, we provide the network throughput under the multi-source scenario. We select 2 source nodes that are 4 hops away from the base station, and show the number of packets received by the base station within the simulation time (50s) against the reporting interval. Similar to the simulation results, ECR-MAC's throughput is more than twice of STEM due to its efficient contention handling mechanisms. The fully-active MAC, which does not use duty cycles, naturally has the highest throughput. It should also be noted that from the whole packets sent from source nodes, only a small portion of them can eventually reach

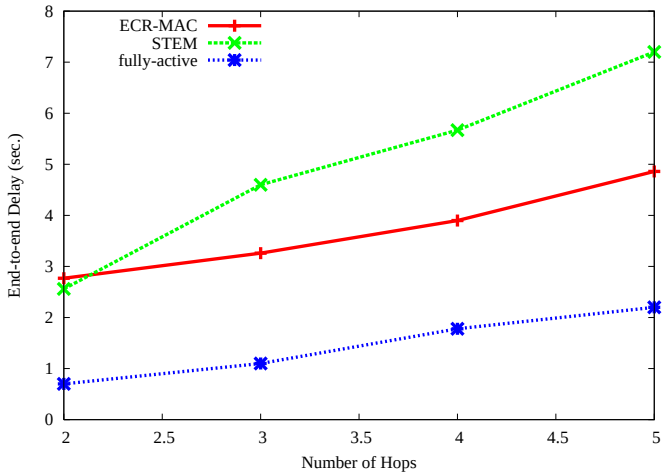


Figure 2.19: End-to-end Delay in MOTE Test-bed

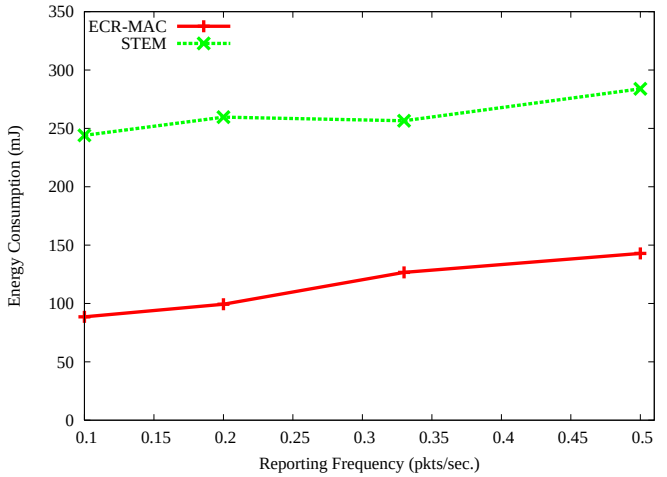


Figure 2.20: Energy Consumption in MOTE Test-bed (Single-source)

the base station (even for the fully-active MAC) with the increase of reporting frequency. This is because the poor link quality leads to significant packet losses and causes retransmissions that waste bandwidth. It has been reported in [94] that there usually exist a "*gray area*" within the communication range of a node. Receivers in the gray area are likely to experience unstable packet reception: some nodes have near 90% successful reception rate, while some others see less than 50% reception rate. The gray area's extent is also big: it can be almost a third of the communication range in some environments. If each node only identifies one fixed forwarder (we assume the widely-used shortest path routing is utilized as in [67]), the network throughput will be significantly impaired due to the poor link quality. This problem becomes even more

severe when duty cycles are used, since the time window a node spends to receive message is significantly reduced by a factor of ($1/\gamma$, γ denotes the duty cycle) compared to fully-active protocol. In the MOTEs test-bed experiments, we observed an advantage of ECR-MAC for handling the poor-quality links and gray area: ECR-MAC allows each node to identify multiple potential forwarders and dynamically choose the one according to their wakeup times and current network conditions, so the links with poor communication quality can be automatically depressed during their competition for becoming real forwarders. This feature of ECR-MAC makes it more suitable to be applied in wireless sensor networks where the network condition and link quality are not guaranteed. In Figure 2.22, we compare the energy consumption of ECR-MAC and STEM (we skip the fully-active MAC that consumes about 10 times higher energy than ECR-MAC). Clearly, ECR-MAC is more energy-efficient to handle contention. However, due to the restriction of test-bed network in which each node can identify limited potential forwarders (less than 3), ECR-MAC's energy-efficiency improvement is lower than our simulation results where we assume a large number of nodes are densely deployed in a large scale network.

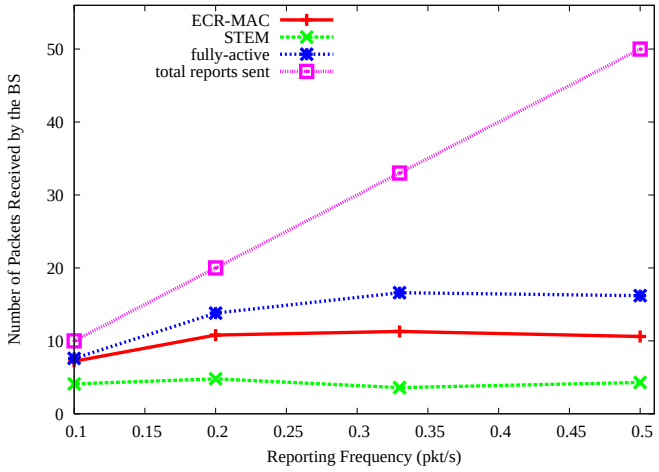


Figure 2.21: Network Throughput in MOTE Test-bed

2.5.3 Differential Duty Cycle Assignment

Our differential duty cycle assignment approach can be generally applied on WSN MAC protocols that deploy duty cycles, as long as the protocol does not depend on all nodes employing a uniform duty cycle. We incorporated our differential approach into ECR-MAC which has been shown to achieve better energy

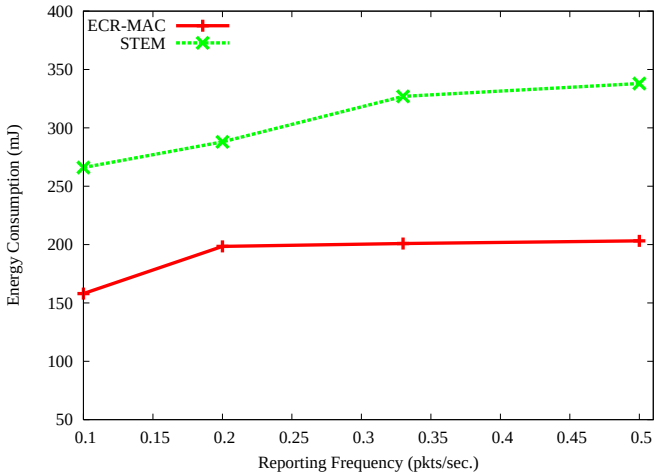
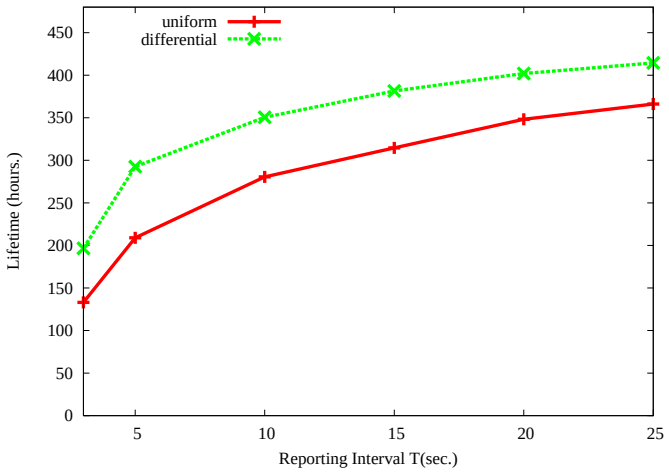


Figure 2.22: Energy Consumption in MOTE Test-bed (Multi-source)

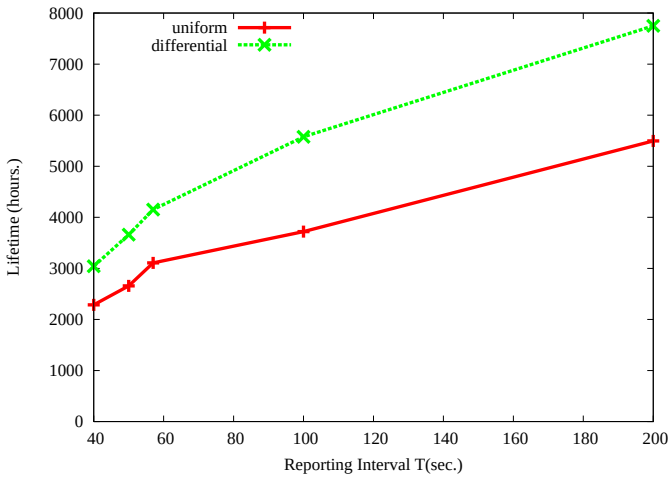
efficiency and shorter end-to-end delay than others in the previous section, and compare our differential approach to uniform approach (as in the original ECR-MAC) in terms of lifetime, energy consumption, and end-to-end delay. To apply our differential approach, we add necessary code in ECR-MAC to let each node be aware of its neighbors' wakeup times.

As the simulations for ECR-MAC, we evaluate our differential approach by following both low and high parameters in Table 2.1 and 2.2, with the aim to show its performance under different network conditions. Since energy hole problem becomes more severe in a large network, we enlarge the network area to be a 1/4 circular area whose radius is 150m and a base station is located at the center of the circle. We randomly distribute 1200 nodes in this area, and each node's transmission range is set to 15m, so the whole WSN is composed of 11 coronas (including the base station at corona 0). We assume all nodes have the same initial battery capacity of 10.8×10^3J that is "2/3 of the capacity of two AA batteries" [31]. According to simulation results in the previous section, in the uniform approach we set ECR-MAC's duty cycle $\gamma = 2\%$ since it achieves a good balance point between energy-efficiency and delay as observed in Figure 2.12. When an event happens, we assume it will be detected by multiple nodes that are close-by, and we randomly select 8 from them to report the event to the base station by sending 5 reports within 1 second after detecting the event. Thus, these results also account for the spatially-correlated contention problem prevalent in WSN. Each data point shown is the average result of 20 independent simulations with different seeds.

In Figure 2.23, we provide the lifetime of uniform and differential approaches for ECR-MAC against



(a) High Parameter

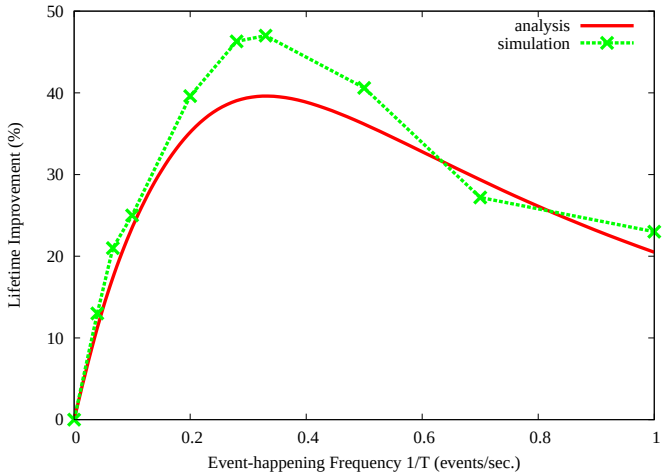


(b) Low Paramenter

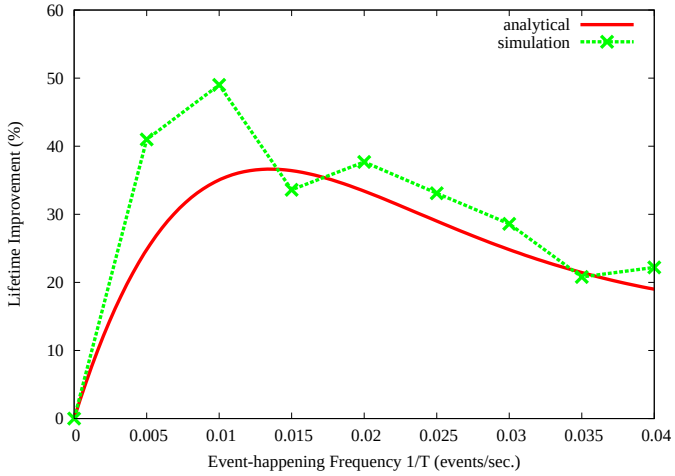
Figure 2.23: Lifetime Comparison between Differential and Uniform Approaches

event-occurring interval T. With the increase of T, the lifetime of both uniform and differential approaches increase since nodes spend less energy for reporting events. For both high and low parameter setups, the differential approach, which addresses the energy hole problem, always achieves longer lifetime than the uniform approach.

To provide a clear view of the benefit provided by differential approach, we show differential approach's lifetime extension ratio over uniform approach in Figure 2.24, in which we also provide the analytical lifetime extension by using equation (2.15) for comparison. It should be noted that for clarity we use event-happening frequency (i.e., $1/T$) instead of T in the x axis. We can observe that for both parameter



(a) High Parameter



(b) Low Paramenter

Figure 2.24: Lifetime Extension by Using Differential Approach

setups the simulation and analysis results are close, and both show that our differential approach extends network lifetime over 20% for most event reporting frequencies. It is also interesting to note that there exist a peak lifetime extension (47% for the high parameter and 50% for the low parameter) at a certain reporting frequency, as suggested by equation (2.15).

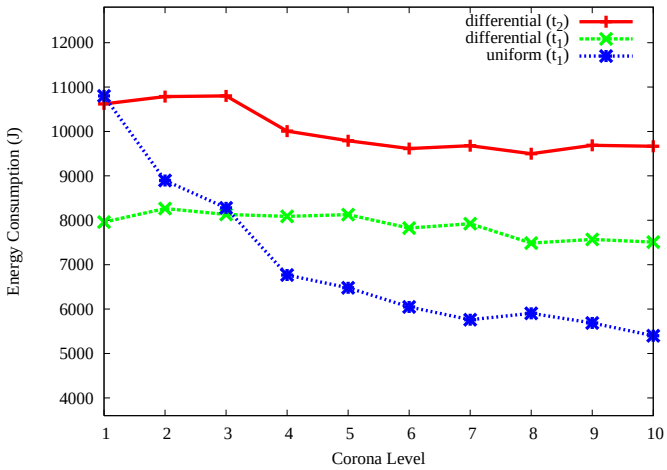
Differential approach extends network lifetime by balancing energy consumption of the nodes in different coronas. In Figure 2.25 we set the event-happening interval $T = 10$s/50s for high/low parameters, and compare energy consumption in different coronas for differential and uniform approaches after the network lifetime expires (we use t_1 and t_2 to denote the network lifetime for uniform and differential approach,

respectively). For differential approach we also provide nodes' energy consumption at time t_1 for completeness (note that typically $t_2 \geq t_1$). Since lifetime is directly related to the maximum energy consumption of a node in a network, we provide the maximum energy consumption of a node in each corona. As expected, the energy consumption of the uniform approach shows a similar trend as in Figure 2.6, which reflects the energy hole problem, i.e., nodes closer to the base consume more energy. Differential approach, at time t_1, reduces the energy consumption of the nodes closer to the base station and requires the nodes at outer coronas to consume more energy, which is necessary to achieve the same delay as in the uniform approach. At both $t1$ and $t2$, differential approach achieves more balanced energy consumption, which confirms that differential approach can effectively address the energy hole problem. In the differential approach, the nodes closer to the base station consume slightly higher energy than the nodes further away, since the network contention becomes more intensive as approaching the base station, which causes more energy consumption for handling potential packet collisions.

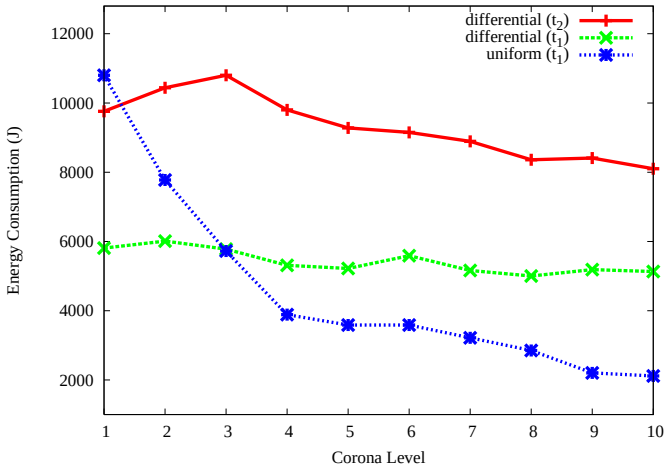
Our differential approach is designed to extend network lifetime without sacrificing delay compared to the uniform approach. In Figure 2.26 we compare the end-to-end delay of these two approaches and the fully-active MAC in two scenarios: a single-source scenario and a multiple-source scenario. In the single-source scenario, we let a single node send one report for an event that happens every 10s/50s for high/low parameters, which shows the basic delay performance. In the multiple-source scenario, we let 8 source nodes send 5 reports for an event that happens every 10s/50s for high/low parameters, and show 10% packets' end-to-end delay, which provides delay performance with contention. We can observe that in both scenarios the differential and uniform approaches achieve similar delays. We also identified the number of reports received by the base station, in our experimentation, to study network throughput. As shown in Figure 2.27, uniform and differential approaches provide similar network throughput. It confirms that our differential approach can extend network lifetime without sacrificing the network performance.

2.6 Summary

In this chapter, we proposed ECR-MAC: a new MAC protocol for wireless sensor networks. ECR-MAC employs DFS (Dynamic Forwarder Selection) mechanism that allows each node to identify multiple potential forwarders to facilitate packet transmissions, which significantly reduces the setup latency and hence



(a) High Parameter

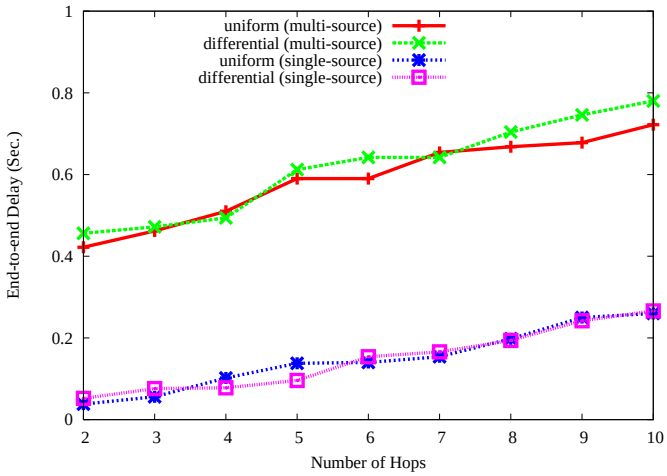


(b) Low Paramenter

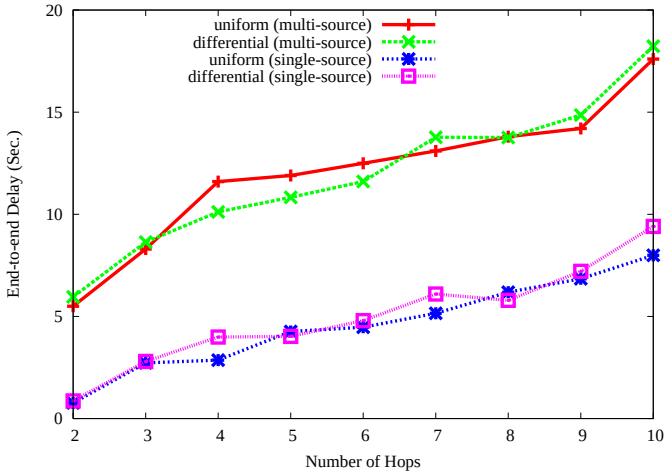
Figure 2.25: Energy Consumption in Different Coronas

improve both energy-efficiency and end-to-end delay without involving additional hardware cost or synchronization overhead. DFS also enables ECR-MAC to efficiently address the spatially-correlated contention in sensor applications by diffusing traffics and dispersing packet transmission times, which improves network throughput and minimizes energy wastage under contention.

As a further study for WSN MAC protocols, we analyzed nodes' energy consumption by both communication traffic and idle listening in a WSN, and addressed the energy hole problem by assigning different duty cycles to nodes at different distances from the base station. We combined this differential approach into ECR-MAC, which not only reduces the energy wastage of idle listening, but also balances the energy



(a) High Parameter

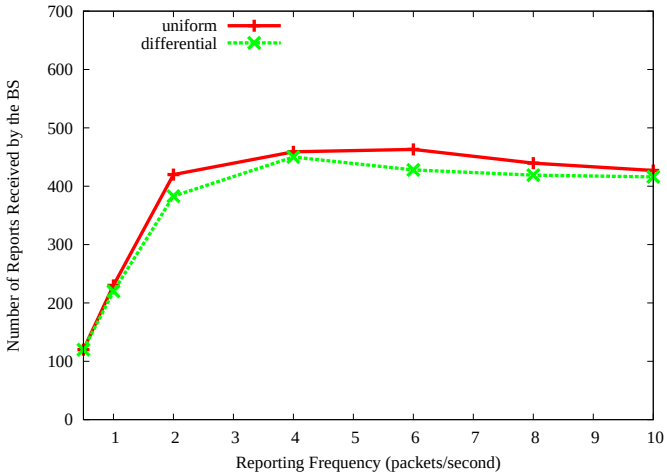


(b) Low Paramenter

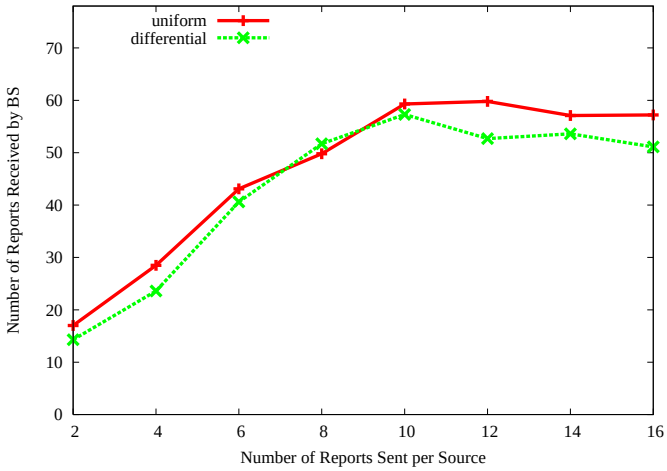
Figure 2.26: End-to-end Delay

consumption across the network, and hence leads to further lifetime extension without sacrificing network performance.

We evaluated ECR-MAC by using both ns2 simulators and MICA2 motes testbed. Both results showed that ECR-MAC achieves shorter end-to-end delay, higher throughput and saves more energy in both single-source and multiple-source scenarios with spatially-correlated contention, and it scales with the network density very well. We evaluated our differential duty cycle assigning approach by combining it with ECR-MAC. Results showed that when achieving comparable network performance in terms of end-to-end delay and throughput, differential approach can achieve over 20% (up to 50%) lifetime extension compared to the



(a) High Parameter



(b) Low Paramenter

Figure 2.27: Network Throughput

original ECR-MAC that uses uniform duty cycles.

CHAPTER THREE

SLEEP-BASED TOPOLOGY CONTROL FOR WAKEUP SCHEDULING IN WIRELESS SENSOR NETWORKS

3.1 Overview

In the last chapter, we propose ECR-MAC, an energy-efficient MAC protocol which improves network performance and conserves energy with low overhead (i.e., using a single radio and does not require synchronization). Though ECR-MAC can effectively shorten delay in an average case, its delay in the worst case may be large due to its CSMA-based nature – nodes choose their wakeup times independently that deters predicting the end-to-end delay. For some real time sensor applications, a short and bounded end-to-end delay is desirable or even indispensable to ensure their applicabilities. For instance, an indoor intrusion detection system always requires any intrusion to be reported within a short time bound to enable a quick response, while a delayed report is of no use and can cause serious security problems. On the other hand, these applications, besides focusing on the network performance (e.g., delay), still take energy-efficiency as one of the most important designing objectives since a long network lifetime is always desirable to improve a sensor system's applicability. As a trade-off, they usually allow to employ sophisticated protocols with additional overhead (e.g., synchronization, localization) to improve the network performance.

To achieve a short, bounded end-to-end delay with a low duty cycle requires to design efficient wakeup scheduling techniques, which facilitates communications between nodes and the base station by coordinating nodes' wakeup times (synchronization is a prerequisite of wakeup scheduling). Implementing a wakeup scheduling also requires to control topology in a suitable manner to facilitate coordinating nodes' wakeup times, and the desirable topology control technique should fully consider the existence of duty cycles, which is usually ignored by most existing topology control techniques that assume nodes are active all the time and manage network topologies by adjusting nodes' transmission range.

Among existing topology control techniques, sleep-based topology control is the only one that utilizes duty cycles to conserve energy. In typical sleep-based topology control techniques, a subset of nodes are selected for constructing a backbone and communicating, and the remaining nodes turn off their radios and periodically check their eligibility to replace backbone-nodes [84, 18]. So each node generally follows an

active/sleep duty cycle and hence extends the network lifetime. On the other hand, sleep-based topology control techniques ensure retaining a connected network at any time, and sustain network performance, i.e., delay, throughput, etc.

The topology control that we propose to implement our wakeup scheduling belongs to sleep-based topology control since each node follows a duty cycle to conserve energy. However, our topology control differs from existing sleep-based topology control techniques in that nodes are allowed to use much lower duty cycles such that the whole network is not guaranteed to be connected all the time. Using low duty cycles can obviously bring the benefit of better energy-efficiency, however, it is challenging to satisfy a short bounded delay requirement and achieve high throughput without even guaranteeing the whole network to remain connected all the time. Our topology control and wakeup scheduling approach is designed to achieve this goal.

In this chapter, we propose a sleep-based topology control technique to schedule nodes' wakeup times that 1) allow using a small duty cycle (e.g., 1% duty cycle that leads to 99% of energy saving), and 2) maintain network performance for typical sensor applications. In particular, our wakeup scheduling's end-to-end delay can be bounded within a small constant factor of the delay achieved by a fully-active protocol. Furthermore, our topology control is designed to efficiently handle spatially-correlated contention, i.e., it can achieve comparable throughput to a fully-active protocol while minimizing the energy-wastage under contention.

3.2 Related Work

Extensive research has been conducted for topology control techniques. Depending on the tunable parameters used, topology control can be classified into three types [32]: mobile node-based, transmission power adjusting-based, and sleep-based. Mobile node-based topology control techniques are proposed for the WSN consisting of mobile nodes, and they adjust network topology by moving the nodes accordingly [57, 14]. We skip the discussion on mobile node-based topology control since we assume a static node deployment in most sensor applications. Interested readers may refer to [32] for a good review.

3.2.1 Transmission Power Adjusting-based Topology Control

A lot of topology control techniques manage network topology by controlling nodes' transmission power. Most of them attempted to keep a low transmission power, which saves energy for packet transmissions and reduces nodes' degrees [38] and interference [15, 48], while still maintaining certain useful features of the original network, e.g., energy-efficient routes [61, 37, 79], shortest paths [26], etc. These techniques were designed for traditional wireless ad-hoc networks in which arbitrary nodes communicate frequently, so saving transmission power can effectively reduce energy consumption.

Rodoplu *et al.* [61] proposed a localized algorithm to construct Minimum-Energy Communication Network (MECN) for stationary ad-hoc networks that preserves minimum-energy path between each pair of nodes. Later, Li and Halpen [37] proposed an improved version SMECN (for Small Minimum-Energy Communication Network) of MECN by constructing a smaller network that incurs lower link maintenance cost and lower energy consumption. They proved that the necessary and sufficient condition for a subgraph G' to retain minimum-energy path is to let G' contain the minimum-energy subgraph G_{min} that removes all energy-redundant edges. Since creating G_{min} requires a great deal of communication to distant nodes, they designed SMECN to form the sub-optimal subgraph G' that contain G_{min}. They have shown through simulations that SMECN performs significantly better than MECN, while being computationally simpler. Cheng *et al.* studied the *Strong Minimum Energy Topology* (SMET) problem, i.e., adjusting each node's transmission power such that "there exists at least one bidirectional between any pair of sensors (strong-connected) and the sum of all the transmit powers is minimized" [19]. They proved that SMET is NP-Complete and proposed two heuristics that provide near-optimal results. Wang *et al.* proposed a localized Shortest-Path-Tree (SPT) based algorithm to construct an energy-efficient topology for wireless ad-hoc networks [79]. They assign a weight for each link based on the required transmission energy, and retain the links that contribute to the minimal energy consumption between two nodes. The generated topology was shown to achieve lower total power consumption (defined as the sum of all nodes' required transmission power) than that of SMECN, and also bound the power stretch factor compared to the minimum-energy path between any two nodes.

Besides keeping the minimum-energy path to achieve energy-efficiency, there are other metrics that have been considered in topology control techniques. For instance, it is desired to bound the node degree to a

constant, so as to prevent communication and energy bottlenecks. In [38], Li *et al.* proposed a minimum spanning tree based topology control algorithm (denoted as LMST), in which each node first collects the location information of nodes within its maximum transmission range, and then builds a local minimum spanning tree independently. The generated topology preserves the network connectivity, has a bounded node degree of 6, and can be transformed into one with only bidirectional links. However, bounding the node degree is naturally opposite to retaining the minimum power path, and it has been shown that no network with a constant node degree contains the minimum power path for any pair of nodes [80], so researchers addressed these two issues by identifying a good balancing point. In particular, the idea of *power spanner* was proposed, i.e., the energy consumption of the energy-optimal routes in a subgraph G' is at most a constant factor away from that of the energy-optimal routes in G [66]. Wang and Li are the first ones to propose a localized algorithm to build a degree-bounded power spanner subgraph both in a centralized and distributed way [89]. However, the node degree can reach 25 in the worst case, and the topology construction requires considerable message overhead. Later, Song *et al.* designed a power efficient topology with lower node degree bounds and less communication costs [68].

While most earlier topology control techniques attempted to address the energy-efficiency issues by minimizing the total energy consumption of the whole network, recently there have been a few works targeted toward the energy balance issue in topology control. Baek *et al.* [12] proposed a power-aware topology control algorithm in which each node determines the communication links based on its neighbors' location and residual energy, and they showed the resulting topology can achieve higher lifetime than LMST and SPT. Dong presented a formal analysis of a variety of network lifetime maximization problems in two different models: *time-based model* in which nodes spend most of their energy on idle-listening while rarely transmitting packets; and *packet-based model* in which nodes communicate frequently such that the energy spent for packet transmission dominates the network lifetime [24]. They provided polynomial time algorithms for tractable problems and NP-hardness proofs for intractable problems.

There also exist several topology control techniques that achieve other objectives. Gao *et al.* proposed a Restricted Delaunay Graph (RDG) routing graph for wireless ad-hoc networks [26]. The most salient feature of this graph is its bounded maximum route length, i.e., "between any two nodes there exists a path in the RDG whose length, whether measure in terms of hops or Euclidean distance, is only a constant times the

optimum length possible"[26]. This feature can help to control the end-to-end delay of packet transmissions.

3.2.2 Sleep-based Topology Control

All the above mentioned topology control techniques focus on conserving transmission power, however, for typical WSN applications, nodes usually spend the majority of time for monitoring and idle listening, while only spending a small portion of time for communicating, so the most effective way to conserve energy is to turn nodes' radios off as much as possible [84, 24, 67, 43, 83]. Based on this consideration, sleep-based topology control techniques have been proposed, which exploit node redundancy to save energy while maintaining the network performance. SPAN [18] is one of the first sleep-based topology control techniques for wireless ad-hoc networks. In SPAN, only a subset of nodes called *coordinators* will remain active. The remaining nodes turn off their radios and periodically check their eligibility to become coordinators based on the information of current coordinators and neighbors. SPAN was shown to extend network lifetime by twice and keep the network performance by providing similar packet delivery rate and delay as ordinary networks. GAF [84] is another sleep-based topology control technique that reduces energy consumption by identifying nodes that are equivalent from a routing aspect and turning off unnecessary nodes. The nodes with higher residual energy will have greater priority to become active nodes. GAF can extend network lifetime by 4-6 times while maintaining the communication fidelity, e.g., delay and throughput.

3.2.3 Wakeup Scheduling

To further conserve energy and extend network lifetime, researchers have designed wakeup scheduling schemes and MAC protocols which typically allow using a much lower duty cycle (e.g., $< 10\%$ [42, 23, 85]) than sleep-based topology control. When using a low duty cycle, however, wakeup scheduling schemes may no longer sustain network performance since the whole network does not remain connected all the time. So improving the network performance (e.g., delay, throughput) while extending network lifetime becomes an issue in designing wakeup scheduling approaches. Earlier scheduling schemes let each node choose its wakeup time slots randomly, which causes significantly longer delay since each node needs to spend an additional *setup latency* to wakeup its forwarder [67]. To reduce end-to-end delay under low duty cycles, Lu *et al.* proposed a *staggered* wakeup scheduling for convergecast traffic pattern [42], in which a WSN is organized as a convergence tree rooted at the base station as shown in Figure 3.1. The wakeup time slots

of neighboring nodes are staggered as illustrated, e.g., B's wakeup time slot appears before its parent G's wakeup time slot that is followed by J's. Staggered wakeup scheduling eliminates intermediate nodes' setup latencies for a packet transmission except at the source node, so it effectively reduces the end-to-end delay. Recently Keshavarzian, Lee, and Venkatraman proposed a *multi-parent* wakeup scheduling that can reduce nodes' setup latencies and hence shorten the delay by allowing each node to maintain two parents that have different wakeup schedules [31]. However, their parent-assigning approach is centralized and NP-complete, which significantly impairs its applicability in WSN where nodes only have limited processing power. In the last chapter we designed ECR-MAC that allows each node to maintain multiple potential forwarders to shorten setup latency and relieves spatially-correlated contention by diffusing traffic. However, since CSMA-oriented ECR-MAC does not benefit from synchronization, its delay in the worst case may be large.

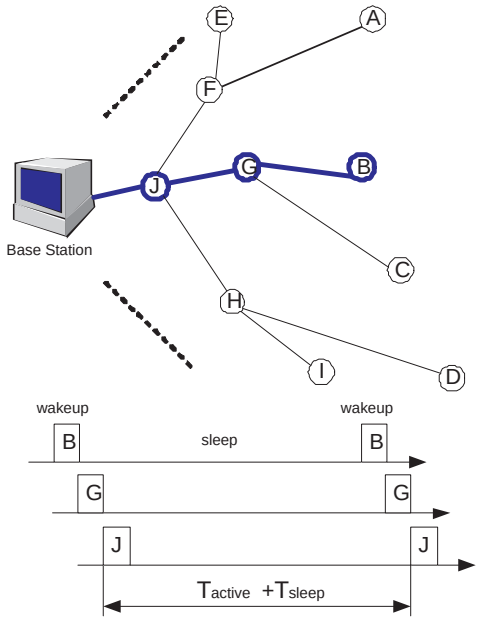


Figure 3.1: Staggered Wakeup Scheduling Illustration

3.3 Motivation and Requirements

Our sleep-based topology control is designed to support an efficient wakeup scheduling that achieves significant energy-savings, low and bounded end-to-end delay, and high throughput under spatially-correlated contention. Since wakeup scheduling schemes are usually incorporated in a MAC [42], our topology control can also effectively improve MAC protocol's performance. Before presenting our topology control

technique, we first describe the requirements of wakeup scheduling that our topology control is designed to support.

3.3.1 Energy-efficiency

In typical sensor applications where events do not happen frequently, the energy consumption is in general determined by the duty cycle, so a low duty cycle is desirable for conserving energy. However, a long T_{sleep} can adversely affect delay and throughput, so a desirable wakeup scheduling should address T_{sleep}'s impact on the delay and throughput.

3.3.2 Spatially-correlated Contention

We observed that a fundamental factor that degrades throughput under contention is the competition for packet forwarding. Traditional protocols usually use a data gathering tree for convergecast (as shown in Figure 3.1), which is naturally prone to contention and collisions. An effective method to improve network throughput is to reduce such competing for common forwarders. As in the ECR-MAC, our wakeup scheduling allows each node to maintain multiple *potential forwarders* for packet forwarding, which can diffuse traffic, relieve contention, and hence improve throughput.

3.3.3 End-to-end Delay

Relieving contention by deploying multi-parent scheduling is helpful to improve throughput, but it does not necessarily lead to a small and bounded delay due to the unpredictable setup latency T_{setup} in each hop (though it can shorten T_{setup} in an average case as shown in ECR-MAC). In order to provide a short, bounded end-to-end delay, we further require that in the multi-parent scheduling each node's parents' wakeup times should be evenly distributed within a $T = T_{active}+T_{sleep}$ period. To fulfill this requirement needs to leverage topology control and node synchronization as opposed to letting each node choose their wakeup times independently (as in ECR-MAC), and the benefit of satisfying this requirement is to provide a guaranteed minimal setup latency in each hop. Assume a sender *A* has *k* potential forwarders, whose wakeup time slots are numerated as $\{t_1, t_2, \cdots, t_k\}$ within T, and we assume the intervals between their wakeup times are $\{v_1, v_2, \cdots, v_k\}$, $v_i = |t_{(i+1)\%k} - t_i|, 1 \leq i \leq k$. In Figure 3.2 we illustrate the wakeup time slots of the *k* potential forwarders within T. The cycle in this figure represents a repeated duty cycle, and the arrow

depicts the time flow. It is straightforward to show that *A*'s expected setup latency *d* is:

$$d = \frac{v_1^2 + v_2^2 + \cdots + v_k^2}{2(T_{active} + T_{sleep})} \tag{3.1}$$

We can observe that d achieves its minimum value if all wakeup times slots are evenly distributed within T, i.e., $v_x = v_y, 1 \leq x, y \leq k$, and the larger the k, the smaller the d. So an even distribution of multiple forwarders' wakeup times guarantees to obtain the minimal setup latency for the multi-parent scheduling. Further, this even-distribution ensures that no two forwarders of a sender choose overlapping wakeup time slots, which eliminates collisions from potential forwarders and simplifies the protocol design (as observed in Figure 2.12 that forwarder collisions can degrade delay when a high duty cycle is used).

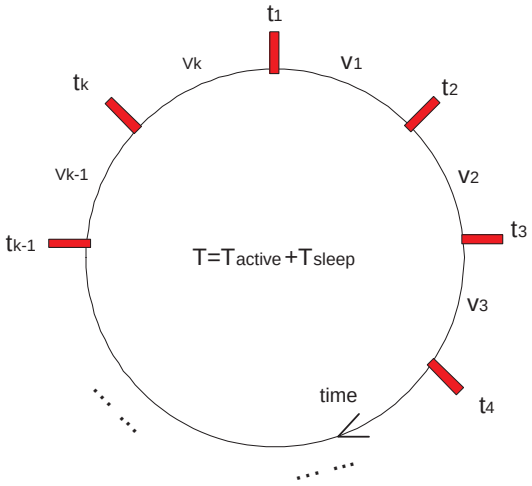


Figure 3.2: Wakeup Times Illuatration

Compared to the multi-parent scheduling that reduces setup latencies in each hop, the staggered wakeup scheduling has been shown to eliminate intermediate nodes' setup latencies along a route to the base station [42, 31]. However, staggered scheduling still has a worst-case of T_{sleep} for the first-hop setup latency, which can significantly prolong the end-to-end delay since T_{sleep} is typically much longer than a packet transmission period. Considering these two wakeup scheduling approaches' complementary features, we believe that allowing each node to have multiple forwarders along with a staggered scheduling can not only relieve contention, but also achieve shorter delay, since it significantly reduces the first-hop setup latency and eliminates setup latencies in following hops.

By summarizing the above considerations, we want to adopt a wakeup scheduling that combines multi-parent and staggered scheduling approaches, which is expected to achieve better energy-efficiency, delay, and throughput. Implementing this combined wakeup scheduling introduces the following challenges for designing topology control and wakeup time coordination techniques:

- Implement Multi-parent Scheduling: since different nodes' neighbor sets can be inter-related and potential forwarders can be shared arbitrarily, the most challenging issue in implementing multi-parent wakeup scheduling is to assign as many multiple potential forwarders for each node as possible and evenly distribute these forwarders' wakeup time slots. The authors in [31] formulated the implementation of multi-parent scheduling as a NP-complete graph coloring problem. They designed a centralized heuristic that can only assign 2 forwarders with different wakeup slots for each node, which significantly restricts its applicability and performance gain. Further, their heuristic allows a node's potential forwarders share the same wakeup slots, which can create collisions of the messages from potential forwarders. In this chapter, we will explore to design a distributed topology control technique to implement multi-parent scheduling such that each node can identify multiple potential forwarders (increase linearly to the network density) and evenly distribute forwarders' wakeup times (we guarantee an even-distributed wakeup time in the network where nodes are uniformly distributed).

- Combine with Staggered Scheduling: combining staggered scheduling with multi-parent scheduling can achieve the performance benefits of shortening delay and improve throughput. However, staggered scheduling usually requires to organize a WSN into a convergecast-tree topology (as in Figure 3.1), which does not coincide with the multi-parent scheduling that forms a mesh topology in which nodes' usually share potential forwarders. Therefore, an efficient topology control technique is desirable to combine these two wakeup scheduling approaches.

We assume that all nodes' local clocks have been synchronized, which is naturally a pre-requisite for coordinating nodes' wakeup time slots [42, 31], and further assume sensor nodes are densely deployed for fault tolerance. Similar to several topology control approaches [84], we also assume that each node is aware of its location, which can be provided by using GPS or other localization techniques [47]. Finally, our topology control is targeted to facilitate convergecast traffic since it is usually the predominant traffic

pattern in typical sensor applications [42]. It also benefits forward direction traffic as described later. For ease of presentation, we first describe our topology control for supporting multi-parent wakeup scheduling in WSN with different node deployment (e.g., concentric circular, uniform, and random), and then present the method to combine multi-parents with staggered wakeup scheduling.

3.4 Topology Control for Multi-parent Wakeup Scheduling

3.4.1 Concentric Circular Deployment

First, we consider multi-parent wakeup scheduling in a very simple WSN where all nodes are uniformly distributed as a set of cycles centered at the base station, as illustrated in Figure 3.3. Though this WSN is over-simplified and ideal, we start with it as a motivating basis for a realistic WSN. We use r to denote the distance between neighboring circles, and assume $r < R < 2r$, where R is a node's transmission range. Each node will take the neighbors that are one level closer to the base station as its potential forwarders. As shown in Figure 3.3, the node n_M^i has 5 potential forwarders. Since in each circle nodes are uniformly distributed, the nodes at the same circle should have equal number of potential forwarders. Assume, in circle M, each node has K_M potential forwarders, and further assume in the circle $M-1$ there are totally N_{M-1} nodes, and N_{M-1} is divisible by K_M. We assign an index for each node: in a circle M, the index i of a node n_M^i is determined by its angle α to the base station as shown in Figure 3.3, and we call α as n_M^i's *position angle*. For two nodes n_M^i and n_M^j, we have $i < j$ iff $\alpha_i < \alpha_j, 0 \leq \alpha_i, \alpha_j < 2\pi$.

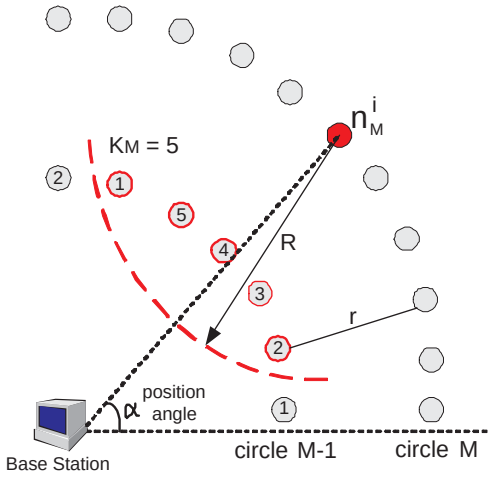


Figure 3.3: Concentric Circular Deployment

To evenly distribute each n_M^i's K_M potential forwarders' wakeup times within a $T = T_{active} + T_{sleep}$, we divide T into K_M portions. A node n_{M-1}^j in circle $M-1$ selects its wakeup slot t_i from the set $\{1, 2, \cdots, K_M\}$, which represents the time offset within T.

Definition 3.1: *A node n_M^l's potential forwarders are **compacted organized** iff. for any two of n_M^l's potential forwarders n_{M-1}^i and n_{M-1}^j ($i < j$), there is no other nodes n_{M-1}^k such that $i < k < j$ and n_{M-1}^k is not n_M^l's potential forwarder.*

Essentially compacted organization requires that there is no "hole" between a node's forwarders.

Lemma 3.1: *In a WSN with concentric circular deployment, each node's potential forwarders are compacted organized.*

Proof: We prove it by contradiction: assume there is a node n_{M-1}^k in the ring $M-1$ that is not a node n_M^l's potential forwarder, and n_M^l has two potential forwarders n_{M-1}^i and n_{M-1}^j such that $i < k < j$. It is easy to see that $|n_{M-1}^k, n_M^l| < max\{|n_{M-1}^i, n_M^l|, |n_{M-1}^j, n_M^l|\}$, where $|n_{M-1}^k, n_M^l|$ denotes the distance between n_{M-1}^k and n_M^l. So n_{M-1}^k must be a neighbor of n_M^l. According to our potential forwarder assigning policy, n_{M-1}^k should be a potential forwarder of n_M^l, a contradiction. So any node's potential forwarders are compacted organized. □

Definition 3.2: *The nodes in a circle $M-1$ is **sequentially synchronized** iff for any two nodes n_{M-1}^i and n_{M-1}^j in circle $M-1$, they choose the same wakeup time slots (i.e., $t_i = t_j$) iff $i \equiv j \mod K_M$.*

An example of sequentially synchronization is shown in Figure 3.3, in which $K_M = 5$ and the number inside a node in the circle $M-1$ denotes its wakeup time slot. It can be observed that nodes' wakeup time slots appear repeatedly with an interval of K_M.

Theorem 3.1: *In a WSN with concentric circular deployment, the necessary and sufficient condition for each n_M^i's potential forwarders' wakeup time slots to be evenly-distributed within T is that all the nodes in the circle $M-1$ are sequentially synchronized.*

Proof: First we prove the sufficient condition. Assume the nodes in the cycle $M-1$ are sequentially synchronized. Since the nodes in each cycle are uniformly distributed and each node choose all its neighbors that is in the cycle one level closer to the base station as its potential forwarders, each node n_M^l in the cycle *M* has the same number of potential forwarders K_M. Since each node n_{M-1}^i chooses its wakeup time slot from the set $\{1, 2, \cdots, K_M\}$, we only need to show that for each n_M^l, all of its potential forwarders choose

different wakeup time slots, and we prove it by contradiction. Assume n_M^l's two potential forwarders, n_{M-1}^i and n_{M-1}^j choose the same wakeup time slot, i.e., $t_i = t_j$. Without loss of generality we assume $i < j$. Since all nodes in the cycle *M-1* are sequentially synchronized, we have $i\%K_M = j\%K_M$, and hence $j - i \geq K_M$. According to Lemma 3.1, we know each node's potential forwarders are compacted organized, so the number of n_M^l's potential forwarders is larger than K_M, a contradiction. So the sufficient condition holds.

Next we prove the necessary condition, i.e., assume each n_M^l's potential forwarders' wakeup time slots are evenly distributed, then the nodes in the cycle $M-1$ are sequentially synchronized. We prove it by contradiction. Assume the nodes in the cycle $M-1$ are not sequentially synchronized, then we have at least a pair of nodes, n_{M-1}^i and n_{M-1}^j, choose the same wakeup time slots (i.e., $t_i = t_j$), but $i\%K_M \neq j\%K_M$. Without loss of generality, we assume $i < j$ and *i, j* are the closest indexes among all the node pairs that violate the sequentially synchronization. Since t_i and t_j must be selected from $\{1, 2, \cdots, K_M\}$, we have $j - K_M < i < j$. So we can always find a node n_M^l that takes both n_{M-1}^i and n_{M-1}^j as its potential forwarders, which means n_M^l's potential forwarders' wakeup time slots are not evenly distributed, a contradiction. So the necessary condition holds. □

Theorem 3.1 provides a method for implementing multi-parent wakeup scheduling in a WSN with concentric circular topology, which will be extended to realistic WSN in the following.

3.4.2 Uniform Deployment

We now relax the restriction of the concentric circular deployment. In Figure 3.4 we illustrate a WSN that we will study in this section, in which nodes are uniformly distributed with the base station at the center. Compared to the WSN with concentric circular deployment as in Figure 3.3, this uniform WSN is closer to a realistic WSN, but adds more challenges for implementing multi-parent wakeup scheduling.

We observe that it is possible to organize a uniform WSN into a set of rings, as illustrated in Figure 3.4, and exploit similar idea as in the concentric circular deployment to implement the multi-parent scheduling in this uniform WSN. We let the nodes n_M^i in the ring M choose potential forwarders from the ring $M-1$ (under the control of certain forwarder selection rules as described later). Different from the circles in Figure 3.3, each ring in Figure 3.4 has a width μ, which has a great impact on the performance of the multi-parent scheduling scheme, since it determines the number of potential forwarders each node can have and the

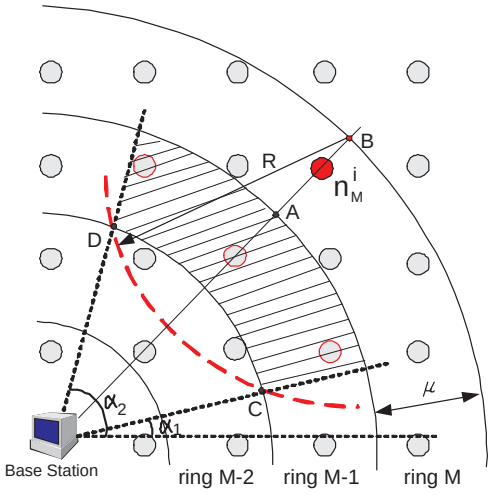


Figure 3.4: Uniform Deployment

route length (in hops) for packet transmission. To let each node have sufficient potential forwarders, we set $0 < \mu < R/2$. However, μ should not be arbitrarily small, which can enlarge the route length unnecessarily and increase the end-to-end delay. In the following, we will first describe the method to choose potential forwarders and evenly distribute forwarders' wakeup time slots, then derive the optimal μ that minimizes the end-to-end delay.

Potential Forwarder Selection

We provide two definitions:

Definition 3.3: *For a node n_M^i, its **outer (or inner) projection** is the point that has the same position angle as n_M^i and is located at the ring M's outer (or inner) edge. In Figure 3.4, the node n_M^i's outer and inner projections are B and A, respectively.*

Definition 3.4: *For a node n_M^i, we call the intersection points of n_M^i's outer projection's transmission radio circle and the inner edge of ring $M-1$ as n_M^i's **bounding points**. Since $0 < \mu < R/2$, each node has two bounding points. In Figure 3.4, n_M^i's bounding points are C and D.*

For each node n_M^i in the ring M, we select its potential forwarders to be the nodes that are located at the ring $M-1$, and whose position angle is bounded by $[\alpha_1, \alpha_2]$, where α_1 and α_2 are the position angles of n_M^i's two bounding points. Note that the selected potential forwarders will be within n_M^i's transmission range. In Figure 3.4, node n_M^i's potential forwarders are the nodes located in the shaded area. Similar

to a WSN with concentric circular deployment, for each node in a ring M, we assign an index i that is determined by its position angle α. If two nodes have the same position angle, we break the tie by letting the node with a bigger ID to have higher index.

Lemma 3.2: *In a uniform and dense WSN, nodes in the same ring have the same number of potential forwarders.*

Proof: Since all nodes have the same transmission range and all rings have the same width, for any two nodes n_M^i and n_M^j in ring M, their forwarders are located in the areas with the same size S_M. Since WSN is uniformly and dense distributed, the number of potential forwarders for a node n_M^i in ring M can be estimated as ρS_M, where ρ denotes node density that is identical overall the network. So nodes in the same ring have the same number of potential forwarders. □

It should be noted that the above proof assumes nodes are densely deployed. In a sparse network the number of potential forwarders can have small variance. To restrict nodes in the same ring to have exactly the same number of forwarders, we require each node to maintain the lowest possible number of forwarders in its ring.

Lemma 3.3: *In a uniform WSN, each node's potential forwarders are compacted organized.*

Proof: Assume a node n_M^l in ring M has two potential forwarders n_{M-1}^i and n_{M-1}^j such that $i < j$. For all the nodes n_{M-1}^k ($i < k < j$), we have $\alpha_1 \leq \alpha_i \leq \alpha_k \leq \alpha_j \leq \alpha_2$, where α_1 and α_2 represent n_M^l's two bounding points. According to the forwarder selection rules, n_{M-1}^k must also be selected as n_M^l's potential forwarder. So each node's potential forwarders are compacted organized. □

Theorem 3.2: *In a uniform WSN, the necessary and sufficient condition for all n_M^i's potential forwarders to be evenly-distributed within T is that all nodes in the ring $M-1$ are sequentially synchronized.*

Proof: According to Lemma 3.2 and 3.3, we can regard rings as special circles in the concentric circular deployment, i.e., nodes in each ring are "projected" to the ring's outer edge as shown in Figure 3.5 (the black nodes are projected to be white ones in rings' outer edges), and the theorem follows. □

Theorem 3.2 enables to use similar method as in the concentric circular deployment to assign potential forwarders and evenly distribute their wakeup time slots within a period T. Based on this, we successfully design a topology control technique that implement multi-parent wakeup scheduling in a uniform WSN, i.e., assigning multiple potential forwarders for each node and evenly distributing their wakeup times. In the

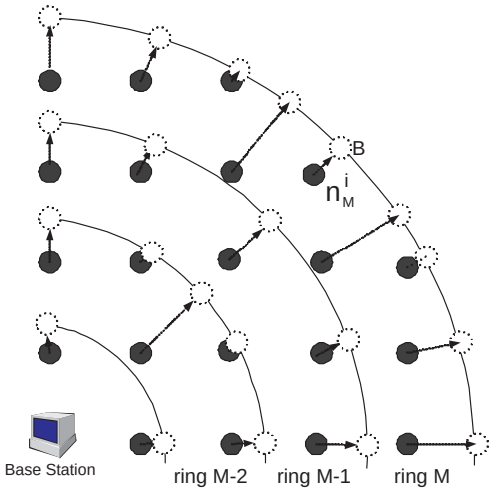


Figure 3.5: Node Projection in Uniform Deployment

following, we will analyze our topology control's performance. In particular, we will derive the number of a node's potential forwarders under our topology control, which helps to determine the optimal ring width μ that leads to the minimal end-to-end delay.

Number of Potential Forwarders

Assume $R = a\mu$, where $a > 2$ is a coefficient, and assume the base station is located at ring 0. So all nodes in the ring $M \leq 2$ can directly reach the base station, and we will only calculate the number of potential forwarders of the nodes in ring $M > 2$. Assume a node n_M^i in the ring M has K_M potential forwarders (K_M is the lowest possible number of potential forwarders for a node in ring M). Since all nodes are uniformly distributed, we have:

$$\begin{aligned} K_M &= \rho\mu^2((M-1)^2 - (M-2)^2)(\alpha_2 - \alpha_1) \\ &= \rho(2M-3)(\tfrac{R}{a})^2 arccos\tfrac{M^2+(M-2)^2-a^2}{2M(M-2)} \end{aligned} \tag{3.2}$$

where ρ is the network density, and α_1, α_2 are the position angles of n_M^i's bounding points as illustrated in Figure 3.4. We set $R = 15m$, $a = 2.4$, and provide K_M against the ring level M for different density ρ (nodes/m^2) in Figure 3.6. It can be observed that K_M remains almost constant for different M, so our topology control assigns a constant K_M for all nodes in a uniform WSN without differentiating its position (ring level), which simplifies the protocol design. In the following, we use K to denote the constant number of potential forwarders in a WSN. We can also observe that K increases linearly with the network density,

which means that our forwarder-assigning method scales with the network density very well.

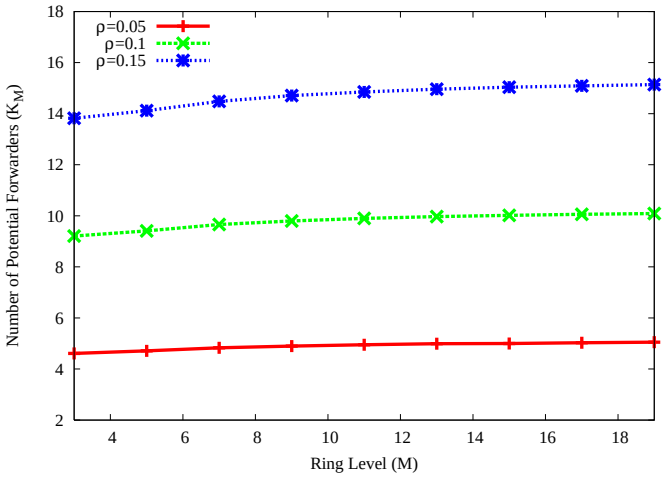


Figure 3.6: Number of Potential Forwarders

Optimal Ring Width

The ring width μ (or the coefficient a) should be carefully controlled since it affects the route length and the number of potential forwarders; both have significant influence on the end-to-end delay. We use d to denote the worst-case end-to-end delay, and d' to denote the delay at each hop. We assume a low duty cycle is used such that $T_{sleep} \gg T_{active}$ and ignore the delay of data transmission since it should be much smaller than the setup latency when a low duty cycle is used (our analysis can be easily extended to include the transmission delay). It should be noted that there will be no collisions among potential forwarders since their wakeup times are evenly distributed. Assume a packet is transmitted from a source that is L meters away from the base station, we have:

$$d = Nd' = \frac{L}{\mu}\frac{T_{sleep}}{K} = \frac{La}{R}\frac{T_{sleep}}{K} \tag{3.3}$$

where N denotes the number of hops, and K can be calculated by using equation (3.2). Our goal is to calculate the optimal a that minimizes d. In Table 3.1 we provide the optimal a against M for different density ρ (nodes/m^2) by numerical computing. Interestingly, the optimal a remains almost constant (around 2.45) for different M and network density, which means different WSN can use the same optimal a that minimizes the end-to-end delay, and the optimal ring width μ is only determined by the transmission range R, i.e., $\mu = \frac{R}{2.45}$.

Table 3.1: Optimal a to Minimize End-to-end Delay

Level (M)	3	5	7	9	11
ρ=0.05	2.494	2.456	2.452	2.451	2.450
ρ=0.1	2.494	2.456	2.452	2.451	2.450
ρ=0.15	2.494	2.457	2.453	2.451	2.450

3.4.3 Random Deployment

In this section, we relax the uniform-distribution assumption and extend our topology control algorithm to realistic WSN whose nodes are randomly deployed.

In a WSN with random deployment, we can use a similar method as in a uniform WSN to assign potential forwarders and synchronize their wakeup time slots. In particular, we organize the whole WSN as a set of rings with the optimal width μ as in a uniform WSN since it minimizes the delay in a general sense, and assign each node an index according to its position angle. We let the nodes in each ring to be sequentially synchronized. For a node n_M^i, we require its potential forwarders to be at ring $M-1$ and within the transmission range of n_M^i's outer projection, but do not bound potential forwarders' position angles as in the uniform WSN due to the irregular network density. Instead, we bound the maximal number of potential forwarders to K calculated by equation (3.2), and minimize the position angle difference between n_M^i and its potential forwarders, i.e., from all nodes that satisfy the potential forwarders selection rules, we first choose the one whose position angle is the closest to that of n_M^i.

It is easy to show that the above rules assign multiple potential forwarders whose wakeup time slots are evenly-distributed within T when used in a uniform WSN, and also ensures that in a random WSN each node's potential forwarders have different wakeup time slots. Due to the irregular node density, our rules do not guarantee that each node can always identify K potential forwarders as in the uniform network. However, we will identify as many forwarders as possible (bounded by K) for each node and ensure forwarders' wakeup time slots are different. So in general, they facilitate the potential forwarder assignment and evenly distribute their wakeup time slots.

Besides random node distribution, realistic networks can also have irregular coverage shapes. The sensor nodes close to network perimeters can not identify as many potential forwarders as internal nodes by nature,

which can prolong transmission delays within the area close to network perimeters. However, since sensor nodes are typically deployed in a large network area with a high density, the number of nodes in the edge areas should be much smaller than the internal nodes. So our topology control allows most nodes in a random network to identify sufficient forwarders with different wakeup times and hence facilitate packet transmissions.

3.5 Staggered Wakeup Scheduling with Multiple Parents

Since we organize a WSN as a set of rings and each ring represents one hop of a packet transmission, the staggered wakeup scheduling can be incorporated by staggering the wakeup time slots of the nodes at neighboring rings. Assume the wakeup time slots of the nodes in the ring M is selected from the set $\{t_1, \cdots, t_K\}$. We set the wakeup time slots of the nodes in the ring $M - n$ from the set $\{t_1 + n\Delta, \cdots, t_K + n\Delta\}$, in which Δ is an estimated packet transmission delay without setup latency and is typically much smaller than T_{sleep} [31]. Since ideally each node can have K potential forwarders, a node n_M^i can always identify a forwarder n_{M-1}^j that wakes up Δ time after it, and hence we achieve the staggered wakeup scheduling with multiple parents. In Figure 3.7 we provide an illustration of the combined wakeup scheduling, in which we assume $K = 3$, so in each ring nodes choose their wakeup times from three possible times (in ring M it is t_1, t_2, t_3). These three possible times are evenly distributed within T. The wakeup times of nodes in neighboring rings will be differed by a Δ. By using this combined wakeup scheduling, for a source at ring M, it takes a worst-case of $T_{setup} = \frac{T_{sleep}}{K}$ in the first hop to wakeup the forwarder at the ring $M - 1$, and takes Δ to forward packet for each of the following hops without setup latencies.

Since we set the ring width $\mu < R/2$, the length of the route generated by our scheme is more than twice the shortest path. To shorten the route length in a random WSN, we allow a node in the ring M to choose forwarders from all rings $M' < M$ by following the same selection rules, which also shortens the setup latency since each node can have more potential forwarders than K. It still ensures that any two potential forwarders for a node have different wakeup time slots since the wakeup time slots of the nodes at neighboring rings are staggered.

Our topology control can be performed in a distributed manner. The base station calculates K using

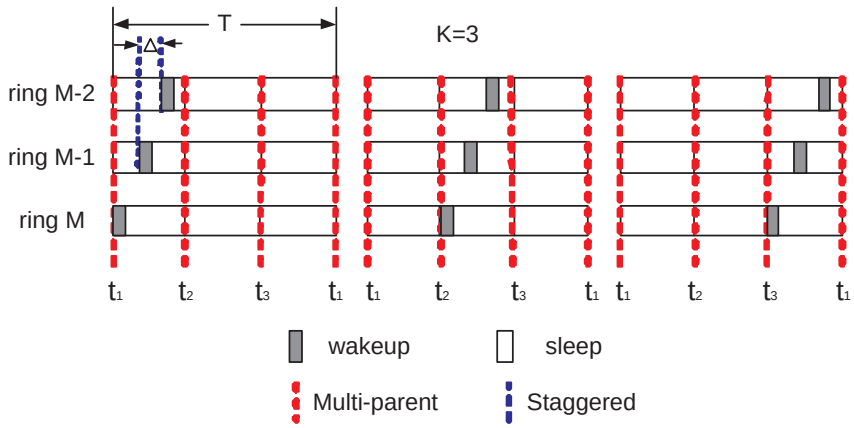


Figure 3.7: Combined Wakeup Scheduling Illustration

equation (3.2) with the knowledge of network density, identifies a baseline with position angle 0, and broadcasts a message that contain K, baseline, and packet transmission delay Δ to all the nodes. So each node can identify its ring level (the ring width $\mu = \frac{R}{2.45}$ can be calculated locally), position angle, the index in its ring, and its potential forwarders. Then each node calculates its wakeup time slot based on its ring level and index.

The previous discussion focused on convergecast traffic. Some WSN applications also require forward traffic from the base station to send commands and queries. These can also benefit from our topology control and the corresponding scheduling, since each node can identify multiple potential forwarders that are closer to the destination to facilitate packet transmissions, which shortens the setup latency and reduces the delay.

3.6 Analysis

Our topology control is designed to support a combined wakeup scheduling and hence improve network performance, so we analyze the effectiveness of our topology control techniques by providing analytical results for its energy consumption and delay estimation, and compare it with a fully-active protocol and three other representative wakeup scheduling schemes: random wakeup scheduling [67], staggered wakeup scheduling [42, 23], and multi-parent wakeup scheduling in which each node is assumed to maintain 2 potential forwarders with evenly-distributed wakuep time slots as in [31].

For simplicity, we assume all nodes are uniformly distributed in a network. We use γ (typically, $\gamma \ll 1$) to denote our approach's duty cycle, and use Δ to denote the delay of a packet transmission. For a typical

sensor application where events happen infrequently, our combined wakeup scheduling can lead to a lifetime extension of $1/\gamma$ compared to a fully-active protocol. A fully-active protocol takes $N\Delta$ to send packets over a N-hop route. If we set $a = 2.45$ (the optimal value to minimize end-to-end delay), our worst-case delay d is:

$$d = 2.45N\Delta + T_{sleep}/K \tag{3.4}$$

Compared to a fully-active protocol, our combined wakeup scheduling prolongs the delay by a factor of $(2.45 + T_{sleep}/(KN\Delta))$. Since K increases linearly with the network density, the impact of T_{sleep} is significantly reduced, and the delay extension factor will be around 2.45 in a dense WSN.

The worst-case delay d of the random [67], staggered [42], and multi-parent [31] wakeup scheduling schemes are:

$$d_{random} = N(T_{sleep} + \Delta) \tag{3.5}$$

$$d_{staggered} = T_{sleep} + N\Delta \tag{3.6}$$

$$d_{multi-parent} = N(T_{sleep}/2 + \Delta) \tag{3.7}$$

We follow the same parameter setup as in [31] to provide analytical comparison: $\Delta = 50$ mSec, the energy consumption for each wakeup period is $E_o = 45\mu J$, the power consumption per unit time is given by $P = \frac{E_o}{T_{sleep}}$, and the initial battery capacity of each node is $E_{total} = 2.4 \times 10^8 E_o$ as in [31]. We set the route length $N = 6$ hops, and choose $K = 8$ potential forwarders for each node in our combined wakeup scheduling scheme (obtained by setting $a = 2.45$ and $\rho = 0.08$ nodes/m^2 according to equation (3.2)). Our analysis is restricted to the convergecast traffic since it is the predominant traffic pattern and enabling immediate event notification is critical for WSN performance [42]. Similar to [31], we provide delay and lifetime for the following two scenarios.

3.6.1 Fixed-power Case

In this scenario, all wakeup schedulings are assumed to use the same T_{sleep} and hence the same P and lifetime. Similar to [31], we assume $T_{sleep} = 2$s, and compare the worst-case delay d (in seconds) of different wakeup scheduling schemes in Table 3.2. We can observe that given a fixed power, our combined wakeup scheduling provides the shortest delay guarantee which is 42.8% of the nearest competing scheme,

i.e., staggered wakeup scheduling.

Table 3.2: Worst-case End-to-end Delay Comparison, in sec.

Combined	Random	Staggered	Multi-parent
0.99	12.3	2.3	6.3

3.6.2 Fixed-delay Case

In this scenario, all wakeup schedulings are required to provide the same delay guarantee. Similar to [31], we assume a common delay bound $d = 1$s and provide their lifetime (in months) in Table 3.3. Our combined scheduling achieves a lifetime extension of three times more than the nearest competing scheme, i.e., staggered wakeup scheduling.

Table 3.3: Lifetime Comparison, in months

Combined	Random	Staggered	Multi-parent
196	11	65	22

3.7 Performance Evaluation

To evaluate the performance of our topology control and the corresponding wakeup scheduling, we divide the evaluation into two parts. In the first part, we evaluate our topology control's effectiveness for implementing multi-parent scheduling (we call it SSI for *Sequentially Synchronized Implementation*) by comparing it with two other implementations: the one proposed in [31] (we call it *2-parent* implementation since it restricts each node to have two parents), and a randomized scheme in which each node randomly chooses its wakeup time slot and selects its neighbors that are one level closer to the base station as potential forwarders (similar to ECR-MAC). In the second part, we evaluate our topology control to support the combined wakeup scheduling scheme by incorporating it into a MAC protocol and comparing its performance with other MAC protocols. For all simulations, we assume 500 nodes with transmission range $R = 15m$ are randomly distributed within a $80m \times 80m$ area, and a single base station is located at the left bottom corner and remains active all the time. According to the results in Table 3.1, we set $\mu = R/2.45$ in our topology control protocol.

3.7.1 Multi-parent Wakeup Scheduling Implementation

We first evaluate our multi-parent wakeup scheduling implementation since it is one of the major tasks that our topology control is designed to support. The objective of multi-parent scheduling implementation is to let each node identify sufficient number of potential forwarders whose wakeup time slots are evenly distributed within a period of $T = T_{active} + T_{sleep}$, and hence minimize the setup latency (refer to Equation (3.1)). So the expected setup latency is a good metric to reflect the performance of multi-parent scheduling implementation. In Figure 3.8, we show the expected setup latency (Equation (3.1)) averaged over all the nodes against network density, and provide a comparison among our implementation (SSI), 2-parent implementation, and random implementation. We set the period of T to 1s, and provided two versions of SSI for comparisons: the first version (single level) is to choose potential forwarders that are one level closer to the base station, and the second version (multiple levels) allows a node to choose potential forwarders that are at least one level closer to the base station.

We can observe that SSI's setup latencies are significantly lower than that of 2-parent implementation, and they decrease in denser networks where each node can identify more potential forwarders (refer to Figure 3.6), so our multiple-parent scheduling implementation scales very well with the network density. The single-level SSI achieves 15%- 23% delay reduction compared to the randomized scheme. Because the irregular network density prevents some nodes identifying sufficient potential forwarders, single-level SSI's setup latency is slightly higher than (within 1.2 to 1.5 times) the optimal value, which is obtained by assuming each node having K_M (Equation 3.2) potential forwarders whose wakeup time slots are evenly-distributed. The multi-level SSI, which allows to choose potential forwarders more than one levels closer to the base station, further reduces the setup latency and even makes it lower than the single-level optimal value in dense networks.

3.7.2 Combined Wakeup Scheduling Evaluation

Our topology control is eventually designed to support a combined wakeup scheduling which incorporates multi-parent and staggered scheduling. In this section, we evaluate the effectiveness of our topology control by incorporating it and the combined wakeup scheduling scheme into a MAC protocol: MS-MAC (for *Multi-parent Staggered wakeup scheduling MAC*), and compare MS-MAC with four MAC protocols ECR-MAC,

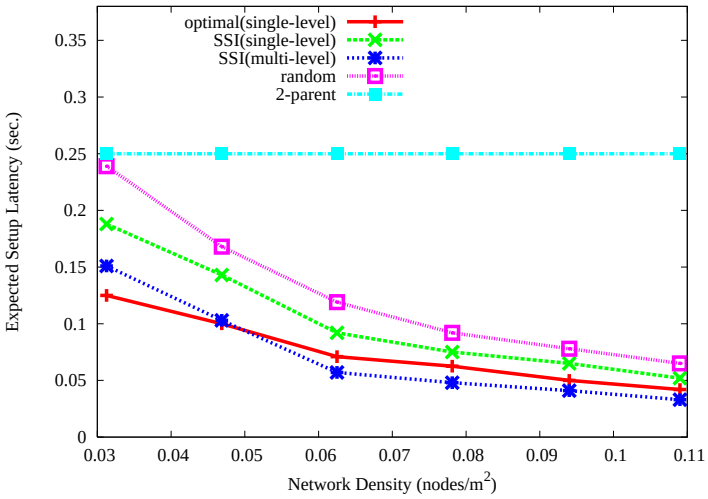


Figure 3.8: Expected Setup Latency

DMAC [42], LEEM [23], and IEEE 802.11, which involves staggered wakeup scheduling (DMAC and LEEM), multi-parent wakeup scheduling (ECR-MAC, without evenly distributing nodes' wakeup times), and fully-active MAC without using duty cycles (802.11). DMAC and 802.11's implementation on ns2 simulator are directly available from [3] and [5], and we implemented LEEM using the ns2 simulator according to [23]. It should be noted that LEEM uses a dual radio setup, so besides the staggered wakeup scheduling, LEEM also uses a pipelining technique [23, 85] to further accelerate packet transmission by saving the delay of wakeup message transmissions. For MS-MAC we set $T_{active} = 2.2$ ms and duty cycle to be 1%. We follow the duty cycles for DMAC and LEEM as in [42] and [23], i.e., 10% and 2.59%, respectively, and use the shortest path routing for DMAC and LEEM as they suggested. For ECR-MAC we set its duty cycle to be 1% and $T_{active} = 2.2$ ms (as in the previous chapter). We follow the bandwidth and radio power consumption parameters listed in Table 2.2, which are the same as in [42].

We conducted simulations in two scenarios: a single-source scenario, in which we compare MS-MAC's basic performance to other protocols, and a multi-source scenario to identify the protocols' performance in a realistic WSN with spatially-correlated contention. Each data point shown is the average of 20 independent simulations with different seed numbers, and we run each simulation for 30 seconds, sufficiently long for a typical event-reporting activity.

Single-Source Scenario

We conducted simulations in the single-source scenario to compare MS-MAC's basic performance to ECR-MAC, DMAC, LEEM, and 802.11. In Figure 3.9, we show the end-to-end delay against the number of hops. It should be noted that we provide the ideal shortest hops in the x axis, which is usually shorter than the routes used by MS-MAC. We include the analytical worst-case delay for MS-MAC and DMAC according to equations (3.4) and (3.6) for comparison, and also show MS-MAC's forward direction traffic delay for completeness. Both MS-MAC and DMAC provide end-to-end delay within their worst-case bounds, and MS-MAC achieves shorter end-to-end delay than DMAC (whose delay is close to what is reported in [42]) and ECR-MAC for convergecast traffic, since MS-MAC combines the staggered and multi-parent wakeup scheduling schemes such that the setup latency is eliminated for intermediate nodes and significantly reduced for the source. MS-MAC's convergecast delay is about twice the fully-active protocol delay, which shows that our topology control technique helps to bound the delay within a constant factor of a fully-active protocol. LEEM, which has two radios and hence can utilize both staggered wakeup scheduling and pipelining technique to accelerate packet transmissions, has a shorter end-to-end delay than MS-MAC for longer (more hops) packet transmissions. MS-MAC's delay increases more quickly than other three protocols (except ECR-MAC that also allows multiple forwarders) as route length increase since it usually requires more hops for packet transmissions. MS-MAC also facilitates forward direction traffic by allowing each node to have multiple forwarders. However, since MS-MAC schedules nodes' wakeup times for convergecast traffic, MS-MAC's delay in forward direction traffic is longer than its convergecast delay.

In Figure 3.10, we show the energy consumption, averaged over all the nodes, against the reporting frequency of a source that is 6 hops away from the base station. We omitted the fully-active protocol's energy consumption which is about 10.6 J and 80 times higher than MS-MAC. Compared to DMAC and LEEM, MS-MAC consumes the lowest energy for both convergecast and forward direction traffic since it uses the smallest duty cycle that dominates the overall energy consumption. MS-MAC consumes slightly lower energy than ECR-MAC (for convergecast) since the combined wakeup scheduling saves energy for waking up forwarders. The forward direction traffic in MS-MAC consumes more energy since nodes' wakeup times are staggered for convergecast.

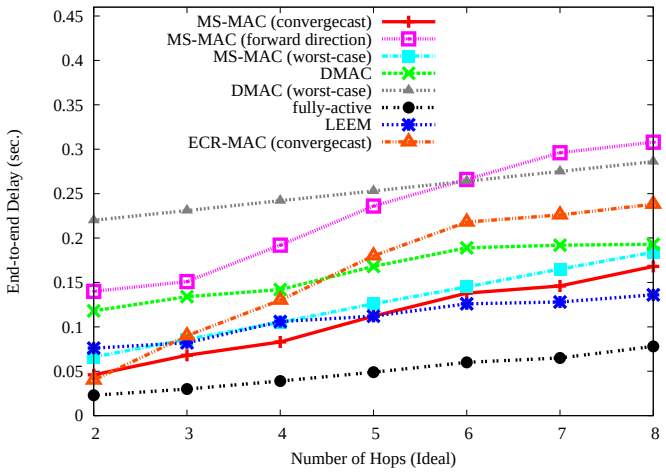


Figure 3.9: End-to-end Delay

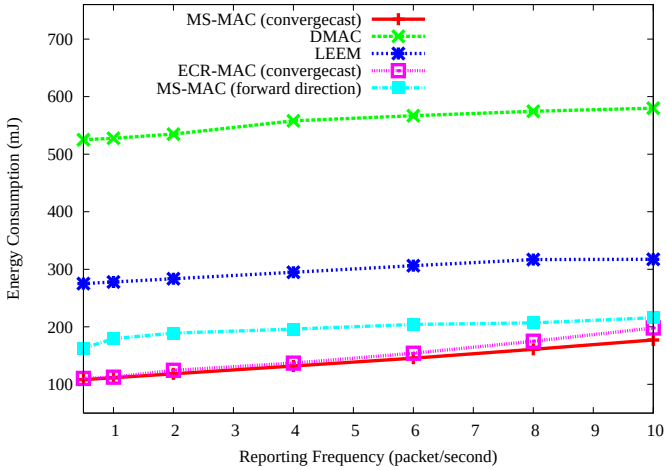


Figure 3.10: Energy Consumption

Multi-Source Scenario

To evaluate protocol performance with spatially-correlated contention, we simulate a realistic scenario in which an event happens at a randomly selected place that is 6 hops away from the base station, and we assume it will be observed by the sensor nodes that are within 10 meters to this place. Similar to the multi-source experiments in the last chapter, we randomly select 8 sources to send reports to the base station within 0.1s after detecting the event, and show the protocol performance against the reporting frequency. In Figure 3.11, we provide the number of reports received by the base station within the simulation period,

which essentially reflects the network throughput since for a typical WSN application nodes collaborate for event detection and reporting. We can observe that MS-MAC achieves about 85% throughput of the fully-active protocol and about twice the throughput of DMAC and LEEM, since its wakeup scheduling can diffuse the traffic from different sources to relieve the contention. MS-MAC's throughput is also higher than that of ECR-MAC since MS-MAC's combined wakeup scheduling enables reports to reach the base station more quickly. ECR-MAC's throughput is higher than DMAC and LEEM since it uses multi-parent wakeup scheduling to relieve contention. DMAC, which introduces additional contention handling techniques (e.g., data prediction and more-to-send (MTS) packet [42]), achieves higher throughput than LEEM. However, DMAC's throughput decreases as the spatially-correlated contention becomes more intense, since it still follows a convergecast tree that is by nature prone to collisions. Further, the intense spatially-correlated contention could matter DMAC's control messages (MTS, ACK) themselves susceptible to collisions.

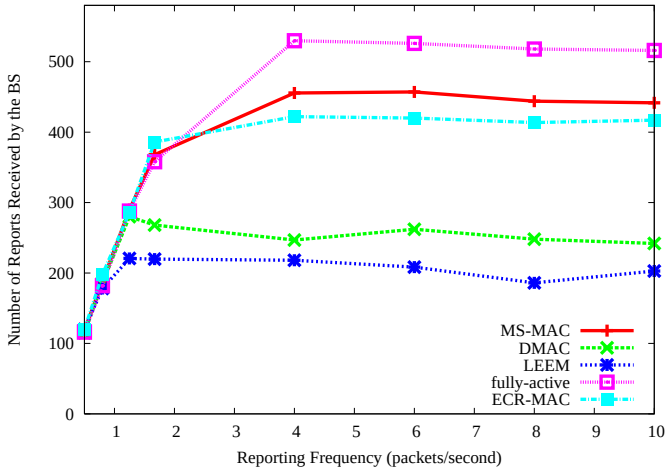


Figure 3.11: Network Throughput

In Figure 3.12, we identified the end-to-end delay of the first 10% reports, which reflects the typical event response time. Since MS-MAC's combined wakeup scheduling not only diffuses traffic and relieves contention but also reduces packet delay by eliminating setup latencies, its delay is comparable to the fully-active protocol and lower than ECR-MAC, DMAC, and LEEM. In Figure 3.13, we show the energy consumption averaged over all nodes. MS-MAC requires the lowest energy consumption since by employing combined wakeup scheduling it minimizes the energy-wastage under contention and uses a low duty cycle

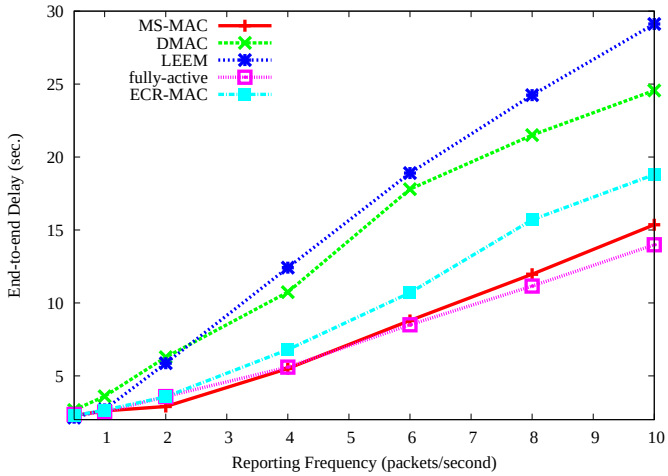


Figure 3.12: End-to-end Delay

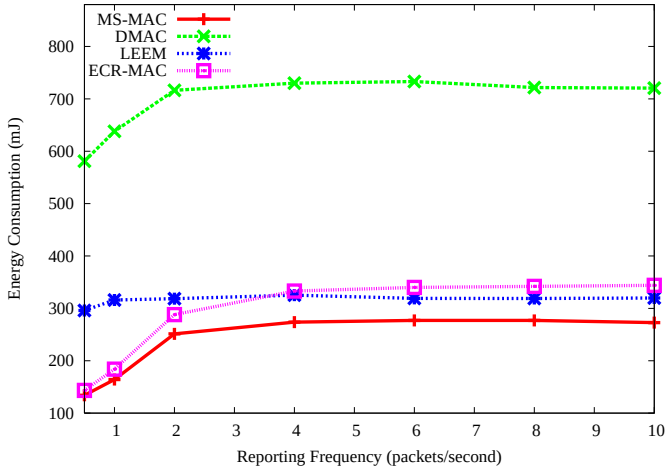


Figure 3.13: Energy Consumption

to conserve energy. The fully-active protocol's energy consumption, not shown in this figure, is about 11.5 J (56 times higher than MS-MAC). To evaluate scalability, we let 8 sources send reports with a frequency of 4 packets/sec., and provide the network throughput against network density (nodes/m^2) in Figure 3.14. MS-MAC shows a better scalability over other MAC since our multi-parent wakeup scheduling benefits from additional forwarders available in a denser network, which relieves contention and facilitates transmissions.

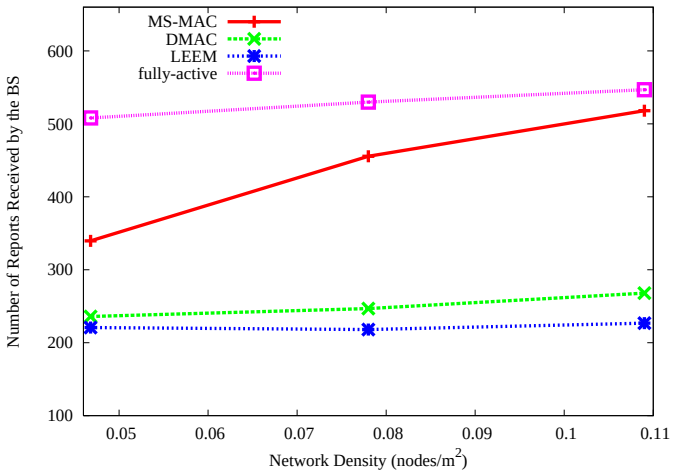


Figure 3.14: Scalability

3.8 Summary

In this chapter, we focused on providing a small, bounded end-to-end delay without sacrificing energy-efficiency for wireless sensor networks by employing wakeup scheduling techniques and leveraging synchronization. To achieve this goal, we design our wakeup scheduling to combine the staggered and multi-parent wakeup scheduling, which leads to significant energy-savings, small and bounded end-to-end delay, and high throughput under spatially-correlated contention. To implement our wakeup scheduling, we proposed a sleep-based topology control technique, in which we organized a network as a set of rings, and carefully choose nodes' forwarders and coordinated their wakeup times such that each node can identify multiple forwarders whose wakeup times are evenly distributed. We also staggered the wakeup times of the nodes at neighboring rings to combine multi-parent and staggered wakeup scheduling, which leads to significant performance improvement.

We implemented a MAC (MS-MAC) to benefit from our wakeup scheduling and topology control, and evaluated our topology control's effectiveness by providing analytical and simulation results, both of which show that our topology control and MAC can bound the end-to-end delay by a small constant (about 2.45) and achieve comparable throughput compared to the fully-active MAC, while consuming much lower energy (by an order of magnitude).

CHAPTER FOUR

ENERGY-AWARE BLUETOOTH SCATTERNET FORMATION AND MAINTENANCE FOR PERSONAL AREA NETWORKS

4.1 Bluetooth Background

4.1.1 Bluetooth Network Stack

Bluetooth is designed as an enabling technology to support personal area networks (PAN). IEEE 802.15 working group has released a standard based on the Bluetooth specifications for wireless PAN. IEEE 802.15 specifies standards on the physical and data link layer of the OSI model, and it has characters such as short-range, low power, low cost, small networks and communication within a personal operating range. We illustrate the relationship among IEEE 802.15 Bluetooth network stack, OSI model and IEEE 802 standards in Figure 4.1.

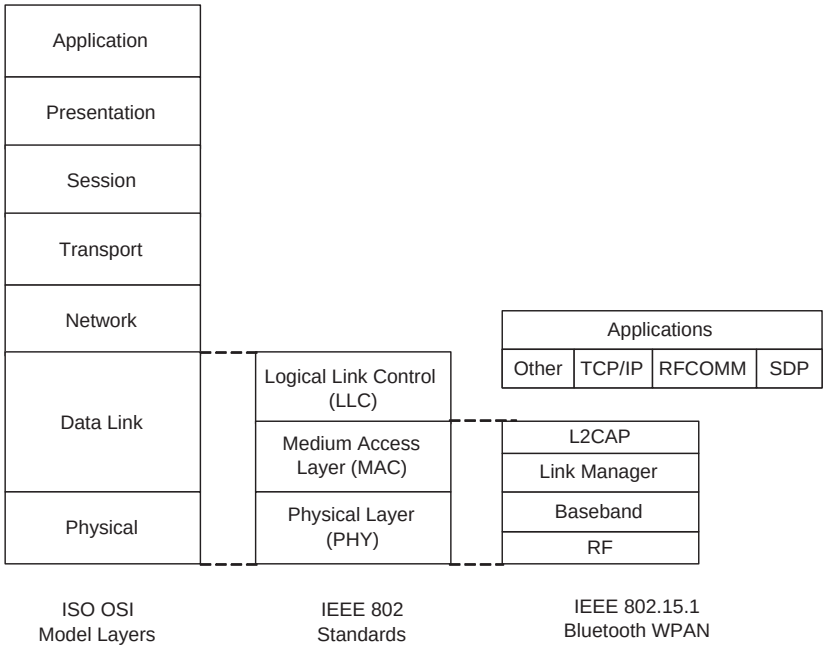


Figure 4.1: Relationship between Bluetooth, OSI model and IEEE 802 Standards

The lowest layer in the Bluetooth network stack, Bluetooth Radio (RF), defines the requirements of the Bluetooth transceiver device operating in the 2.4 GHz ISM band. There are three classes of Bluetooth radios with different output power levels: class 3 radio has a maximum output power of 0 dBm; class 2 radio has

a maximum output power of 4 dBm, and class 1 radio has a maximum output power of 20 dBm. Among them, class 3 radio is the most popular-used, since its weak output power helps to avoid interfering with other type of devices that operate in the same band and saves energy consumption. Class 3 radio support the transmission range up to 10 meters, which is sufficient for typical PAN applications.

Bluetooth Baseband lies on top of Bluetooth Radio, and it also belongs to physical layer. Baseband controls the Bluetooth physical link between devices like link connection and power control. It also manages page and inquiry operation to discover and access Bluetooth devices in the area.

The Medium Access Control (MAC) layer in Bluetooth is composed of Link Manager and L2CAP (for *Logical Link Control and Adaptation Protocol*). The Link Manager controls the link set-up between Bluetooth devices, including security, negotiation of Baseband packet sizes, duty cycle control of the Bluetooth device, and the connection states of a Bluetooth device in a piconet. L2CAP provides the upper layer protocols with connection-oriented services.

RFCOMM is a simple transport protocol that provides emulation of RS232 serial ports over the L2CAP protocol. It supports up to 60 simultaneous connections between two Bluetooth devices. The number of connections that can be used simultaneously in a Bluetooth device is implementation-specific.

SDP (for *Service Discovery Protocol*) provides mechanisms for applications to conduct service discovery and determine the characteristics of available services. SDP uses a request/response model for service discovery and has minimal requirements on the underlying transport protocol.

4.1.2 Bluetooth Architecture

To reduce interference from other electronic devices operating in the same band (e.g., 802.11 WLAN), Bluetooth uses *Frequency Hopping Spread Spectrum* (FHSS) technique by dividing the frequency band into 79 channels, and each device follows a unique frequency hopping sequence that is designed not to have any repetitive patterns over small periods of time. Bluetooth piconet is a group of Bluetooth devices that share a common communication channel (i.e., frequency hopping sequence). Each piconet can contain precisely one master and at most seven active slaves (a piconet may have more slaves which are not active or remain locked in the *parked* mode). Master and slave are only logical states; any device can potentially be a master, a slave, or both (i.e., bridge device as described later).

Due to the employment of frequency hopping technology, Bluetooth is connection oriented, which

means that Bluetooth devices can not communicate unless they explicitly form a piconet. This feature, which is different from IEEE 802.11 in which devices can communicate as long as they are within each other's transmission range, complicates the link setup procedure. To construct a link between two Bluetooth devices (i.e., forming a piconet), both devices need to go through two phases: inquiry and page. In the inquiry phase, a bluetooth device, in the active role, enter INQUIRY state by sending beacons and listening to replies. Another device, in the passive role, perform INQUIRY SCAN to listen to beacons and send replies. Since two devices deploy different frequency hopping sequences, Bluetooth specifications require the device in INQUIRY state to follow a faster frequency hopping rate than the device in INQUIRY SCAN state to accelerate the discovery process. Through control packet exchanges, at the end of inquiry phase the two neighboring devices will obtain address and clock information from each other, and then they perform the page procedure by entering PAGE and PAGE SCAN state, respectively. With the available address and clock information, each device can deduce the other one's frequency hopping sequence, so the page procedure is typically much more efficient than the inquiry procedure. After page procedure the two devices will form a piconet in which the one who went through INQUIRY and PAGE state will become the master and determine the piconet's frequency hopping sequence, and the other device becomes a slave and follows the master's frequency hopping sequence.

In a piconet, devices access channel by following *Time Division Duplex* (TDD) scheme, in which a slave can communicate to the master in the time slot preceded by the time slot in which it was polled by the master, and a slave can not communicate to another slave directly. The TDD scheme ensures a collision-free channel access, but it also reduces the channel utilization ratio, which makes it necessary to restrict the number of devices in a piconet (at most 8) for ensuring each device's data rate.

Since a Bluetooth piconet can only support a small number of active devices, Bluetooth allows to interconnect piconets to form larger networks, called scatternets, by sharing *bridges*. Similar to master and slave, bridge is also a device's logical state. There are two kinds of bridges: Master-Slave (M/S) bridge and Slave-Slave (S/S) bridge. A M/S bridge acts as a master in one piconet and a slave in another, while a S/S bridge acts as slaves in both piconets. But a device can never become a Master-Master (M/M) bridge since a device can only be in charge of one piconet. Scatternet topology has a great effect on the network communication performance. Typically, the number of piconets in a scatternet should be minimized to reduce the

interference among different piconets. Furthermore, since bridge devices need to share communication time slots among multiple piconets and introduce considerable switching overhead, it is preferable to reduce the device degree, i.e., the number of piconets it participates. Finally, a scatternet should be formed quickly to reduce communication interruption.

4.1.3 Energy-Aware Bluetooth Scatternets

In the literature, intensive researches have been conducted to study the issue of scatternet formation and optimization. However, few of them consider energy-awareness as a designing objective. To determine and construct an optimal energy-aware scatternet is an extremely complex problem due to the large number of parameters involved. Marsan et al. have shown a centralized algorithm to determine the optimal scatternet that minimizes the energy consumption of the most congested device in the network while meeting traffic requirements [45]. However, their centralized algorithm is shown to be NP-complete, which is too expensive to be adopted for distributed Bluetooth devices. Therefore, "heuristics for the construction of good (suboptimal) topologies in a distributed fashion are needed" [45].

An energy-aware scatternet should have short links, which enables the sender device to spend less transmission energy if power control is applied [52], and have short diameter, which typically means fewer devices would be involved in data delivery and hence reduce the overall energy consumption [29]. However, since link length and diameter can not be jointly optimized, an integrated design would be critical to obtain a good balance between them. Another requirement for energy-aware scatternet is to assign devices to roles that suit their workload according to energy capabilities. For instance, masters and bridges should be more energy capable than slaves, and in a tree scatternet, higher-level bridges and masters, which typically have higher workload, should be more energy capable than lower-level ones. Besides energy-efficiency, other metrics that affect network performance should also be considered, e.g., the number of piconets, device degree, etc.

Since bluetooth devices are free to move and their various workloads lead to unbalanced energy depletion rates, it is also important to maintain scatternets' energy-aware features after forming a scatternet. The maintaining process, which typically reorganizes the scatternet topology, inevitably interference devices' normal communication, so it is desirable to restrict the reorganization to a small local area that precisely includes the energy-bottleneck, such that the maintaining overhead (e.g., the number of devices involved as

well as the reorganization period) is minimized.

In this chapter, we will study the problem of energy-aware scatternet formation and maintenance: we first identify an ACB-tree (for *Almost Complete Binary tree*) topology for energy-aware scatternets, and then designed a distributed *Energy-aware Multi-hop Tree Scatternet* (EMTS) formation technique based on the ACB-tree topology, which forms energy-aware scatternet by addressing energy-aware requirements in an integrated manner, i.e., shortening link length and diameter, and matching devices' roles to their energy capabilities. In the multi-hop network scenario where two devices can be out of each other's transmission range and can only communicate through intermediate forwarders, EMTS generates the scatternet whose lifetime is significantly longer by up to 7-9 times than that of other algorithms, and it has the desirable characteristics of short link length, short diameter, low number of piconets and low device degree. To maintain the energy-aware features of the generated scatternet, we designed and implemented a localized reorganization technique that effectively extends scatternet lifetime with low overhead, and also addressed the issues of handling dynamic device arrivals and departures.

4.2 Related Work

Scatternet formation has been extensively discussed in the last few years. Existing scatternet formation algorithms can be classified into two types: single-hop algorithms which assume that all Bluetooth devices are within transmission range of each other and multi-hop algorithm that does not require all Bluetooth devices to be in communication range of each other.

4.2.1 Single-hop Scatternet Formation

Earlier works, including BTCP [65], TSF [73], LMS [35] and LSF [91], form scatternets in the single-hop networks, which is a simplified network scenario where all devices are assumed to be within each other's transmission range. In BTCP (*Bluetooth Topology Construction Algorithm*), a single node is elected as a coordinator with the complete knowledge of the whole network, and the coordinator determines scatternet topology as well as notifying all devices involved to create a fully connected scatternet. However, BTCP can support a maximum of 36 devices and can not scale to higher number of devices. TSF (*Tree Scatternet Formation*) forms tree scatternets that are shown to be efficient for packet scheduling and routing. TSF is optimized for those situations that fast device connectivity and short formation delay is the most important

concern. Because of the short formation delay objective, TSF doesn't control the network diameter, number of piconets or node degree. TSF is self-healing and allows any device to join or depart without causing long disruptions in connectivity. Law and Siu proposed LMS [35], a distributed scatternet formation algorithm which achieves $O(\log n)$ time complexity and $O(n)$ message complexity. The number of piconets of generated scatternet is close to be optimal, and the degree of each device is either one or two. Zhang, Hou and Sha [91] formalized the notion of network diameter and node contention, and proposed a *Loop Scatternet Formation* (LSF) algorithm which creates a scatternet with bounded number of piconets, network diameter and node contention.

4.2.2 Multi-hop Scatternet Formation

Later protocols were proposed to form scatternets in the more realistic multi-hop networks. Most multihop protocols include a neighbor discovery phase to have a better control over the scatternet topology [52, 54, 55, 39, 81]. Zaruba *et al.* [90] proposed Bluetree, which forms scatternets based on a process initiated by a node named *blueroot* and repeated recursively until the leaves of the tree are reached. However, selection of the blueroot is not discussed, which would be a costly process in a distributed environment. Petrioli *et al.* proposed two multihop protocols named BlueStar [54] and BlueMesh [55]. Both generate mesh-like scatternets with multiple paths between any pair of nodes. They were shown to quickly converge, and yields scatternets with a lower number of piconets, average route length and number of roles per device. Li *et al.* [39] utilized the *Yao* construction to create a connected scatternet with bounded number of slaves per piconet. However, they require each device to be aware of its geographical position. Wang *et al*. [81] proposed BlueNet, but the generated scatternet is not necessarily guaranteed to be connected, as pointed in [39]. Basagni *et al*. [13] compared four multihop formation algorithms: BlueTree, BlueStar, BlueNet and LSBS, and provided possible solutions to accelerate device discovery process, which enables 90% of neighbors to be discovered within 6 seconds. Sunkavalli and Ramamurthy [72] proposed a mesh scatternet formation algorithm MTSF, which has short delays for formation and new device absorption. Cuomo *et al*. [22] proposed SHAPER-OPT: an integrated approach for scatternet formation and topology optimization in a multi-hop network. SHAPER can create tree-scatternet with very limited formation time and it assures self-healing properties of the network. Zhang and Riley [92] proposed an on-demand scatternet formation and routing protocol which is optimized for wireless sensor applications. However, its on-demand nature

leads to longer scatternet formation delay. Roy *et al.* [63] proposed a topology construction protocol that works in conjunction with a priority-based polling scheme in which a master assigns each of its slave (including bridge) a priority based on the packet arrival rate and the available buffer space in each slave, and poll it accordingly. The topology construction algorithm works in a bottom-up manner in which isolated devices form piconets to scatternets by sharing bridges. To handle new device arrivals and departures, the scatternet needs can be dynamically reconstructed. Drula *et al.* [25] explored the neighbor discovery issue for Bluetooth devices, and proposed two adaptive policies for neighbor communication establishment. The proposed schemes use the recent activity and the location of previous encounters to predict the probability of encountering a nearby device, and dynamically adjusting the Bluetooth parameters to shorten neighbor discovery time and/or conserve energy. Simulation results show that their adaptive algorithms reduce energy consumption by 50% and have up to 8% better performance over a static power-conserving scheme for performing neighbor discovery.

4.2.3 Energy-aware Scatternet Formation

Recently there have been a few attempts to address energy-aware issues in the Bluetooth scatternet. Marsan et al. [45] proposed a centralized algorithm to construct optimal scatternet that maximizes network lifetime by minimizing the most congested device's energy consumption, while meeting network connectivity, capacity and traffic requirements. However, it is NP-complete and too expensive to be applied in practical scenarios: a distributed heuristic to form energy-aware scatternets is in urgent need [45]. Pamuk and Karasan [52] proposed a tree-based energy-efficient scatternet formation algorithm: SF-DeviL, which increases the scatternet lifetime by shortening communication links and assigning more energy-capable devices to be masters, but it does not control the network diameter and incurs a high formation delay. Saginbekov and Korpeoglu [64] proposed a centralized energy-efficient scatternet formation algorithm for sensor networks, which requires the base station to obtain the global knowledge over the network and may not be applicable to general PANs. Medidi *et al.* [46] proposed ETSF to create energy-efficient ACB-tree scatternets in a single-hop network, which has a bounded diameter and device degree, and ensures higher-level devices to be more energy-capable.

4.2.4 Scatternet Maintenance

Scatternet maintenance is another important issue that have been studied intensively. Most maintaining algorithms were designed to keep the scatternet connectivity when handling devices' movements (e.g., joins or departures) [93, 71], while scatternet's energy-efficiency has not been considered. DSOA, the topology maintaining algorithm in [22], can adjust scatternet topology according to selected performance metrics. However, DSOA requires the reorganization over all the devices in the network, which increases its overhead. In SF-Devil [52], reorganization was applied to maintain the energy-efficiency of the scatternet so as to extend its lifetime. Similar to DSOA, SF-Devil requires a global topology update and hence has a high maintaining overhead. The authors in [71] proposed a localized maintaining algorithm. However, it assumes each device to be aware of its position, which may not be available in practical scenarios. The energy-efficient scatternet reorganization techniques proposed in [29] mainly focuses on shortening average path length/diameter, but the link length and the relationship between energy capabilities and device roles are not considered.

4.3 ACB-Tree

4.3.1 ACB-tree Topology and Combination

For the sake of energy-efficiency, we designed an ACB-tree topology for the generated scatternet. An ACB-tree is essentially a complete binary tree with an additional node (termed *handle*) connected to the root as illustrated in Figure 4.2. Recall that a complete binary tree of depth d has $2^{d+1}-1$ nodes, and combining two such complete binary trees into a larger one requires an additional node to act as the new root, which deters designing distributed tree-growing algorithms. An ACB-tree, on the other hand, contains 2^{d+1} nodes, which enable two ACB-trees to combine into a larger one without requiring any additional nodes: two ACB-trees can be easily merged by replacing one edge (from a handle node to its root) with two new edges. In Figure 4.3, edge $R1$-$H1$ is replaced by two edges: $H2$-$H1$ and $R1$-$H2$. Note that combining two ACB-trees of the same height naturally gives an ACB-tree with height one more than the original trees. Further, The height of an ACB-tree remains logarithmic in the number of nodes as long as the height differences of trees being combined are within a constant θ.

Besides enabling an efficient distributed growing process, ACB-tree has several salient features suitable

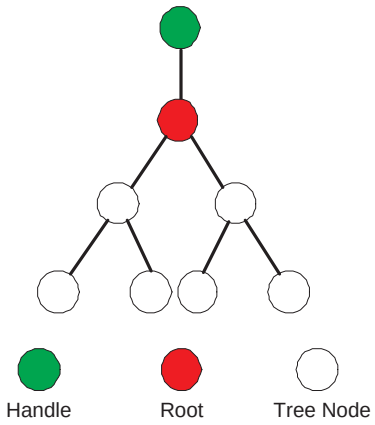


Figure 4.2: ACB-tree

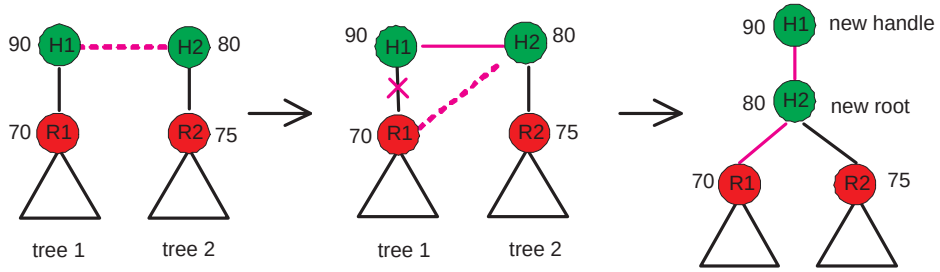


Figure 4.3: ACB-tree Combination

for constructing energy-aware scatternets. First, ACB-trees grow in a balanced manner, so in general it provides the topology with a shorter diameter than other tree scatternets in which two trees can be combined by connecting any two nodes from the two trees (e.g., TSF, SF-DeviL, SHAPER). Second, ACB-trees grow by combining the top of two trees (i.e., handle and root nodes). So by choosing more energy-capable nodes as new handles when combining two trees, ACB-trees' higher-level nodes will have higher energy level and be more suitable to handle the higher workload in a tree topology. In Figure 4.3, the number besides a node represents its energy level. We can observe that after combination the new handle (H1) has higher battery level (90) than H2(80), R1(70), and R2(75). Finally, as a binary tree, the node degree in an ACB-tree is bounded by 3, which effectively prevents communication bottlenecks in a scatternet. Therefore, ACB-tree is a good candidate topology to be applied in forming energy-aware scatternets.

4.3.2 Forming ACB-tree Scatternet in a Single-hop Network

Single-hop Bluetooth network is a simplified scenario of realistic applications since it assumes all devices are within each other's transmission range and provide more choices to control the scatternet topology. For

the sake of clarity, we first use the single-hop scenario to illustrate the basic approach of applying ACB-trees to form energy-aware scatternets, and then extend it to the realistic multi-hop scenario in the following sections. We will represent the energy capability of any device with an integer value between 1 and 99: these energy values are meant to be representative only, and may also be associated with device grade, battery level, etc.

In a ACB-tree scatternet, each node represents a piconet, and the master of a piconet is used to label the node. To minimize the number of piconets, we orient the tree link from child node to its parent (the link is directed from master to slave). Since Bluetooth specification restricts the number of active slaves to 7, and the discovery process of our scatternet formation algorithm requires one free link, each piconet can acquire at most 5 dedicated slaves (the other free link is used to connect piconets). The master of handle node is termed *leader* of an ACB-tree, and the root's is termed *coordinator*.

We design a three-phase technique to form energy-aware scatternets in the single-hop network. In the first phase, devices are collected into piconets, and the most powerful device in each piconet is guaranteed to act as the master. Each device initially acts as the master of its own piconet alternating between INQUIRY and INQUIRY SCAN to discover other piconet masters. Upon discovery, two piconets would combine into a single piconet as long as the number of slaves in the resulting piconet does not exceed 5, and the master with higher battery level becomes the new master. This process terminates when either a piconet acquires 5 slaves or a timeout expires.

In the second phase, piconets (or tree nodes) will be connected to grow ACB-trees. At any time during phase 2, we have ACB-trees and lone nodes (which we regard as a special ACB-tree with a single handle node). For each ACB-tree, we let its leader (the master of its handle) take the responsibility to discover other trees by alternating between INQUIRY and INQUIRY SCAN. Once two leaders establish a connection, they communicate and compare their trees' height. The connection may have to be torn down if their height difference violates the difference restriction θ. Otherwise, they further compare their battery levels, and combine into one ACB-tree by choosing the more energy-capable one as the new leader, as illustrated in Figure 4.3. In general, this combining procedure puts nodes with higher energy levels at higher tree levels to balance energy consumption and improve the scatternet's energy-efficiency. However, it is still possible that a node at low levels has higher energy than upper-layer nodes (we call it an *energy violation*), which is

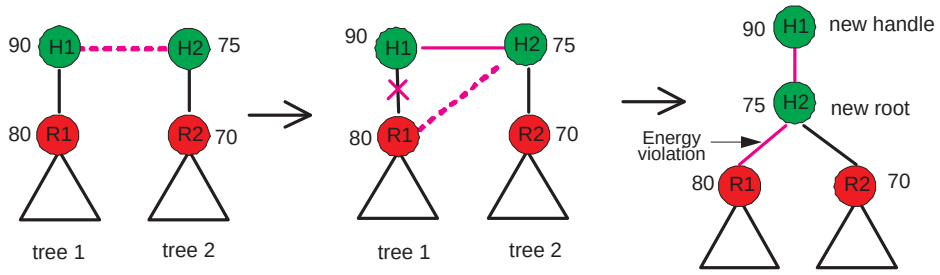


Figure 4.4: Energy Violation Illustration

characterized by the leader of a tree has a lower energy capability than both the coordinator and leader of the other tree, as show in Figure 4.4. We will address this issue by using scatternet maintenance techniques in Section 4.4.3. Phase 2 only combines ACB-trees that differ in height by less than θ and does not guarantee the creation of a single scatternet, and terminates when leader fails to find a suitable tree for merging before a time-out period. In the third phase, the same method as in phase 2 is applied except the height restriction θ is ignored to create a single and connected ACB-tree scatternet.

According to ACB-tree's properties, the generated scatternet should have a low diameter and a bounded device degree. We do not consider link length since in a single-hop network the average distance between two devices are small. Furthermore, the nodes at higher tree levels generally have higher battery level than lower-level nodes. So our technique can form energy-aware ACB-tree scatternets in the single-hop scenario.

4.4 Energy-aware Scatternets in the Multi-hop Scenario

Compared to the single-hop scenario, the multi-hop scenario further complicates the scatternet formation since all devices are not necessarily neighbors, which restricts the choices available to achieve energy-efficiency. Before presenting our multi-hop scatternet formation technique EMTS, we first identify the challenges to form energy-aware scatternets in the multi-hop scenario.

4.4.1 Challenges

Most challenges of forming energy-aware scatternets in a multi-hop scenario stem from the fact that nodes can be out of each other's transmission range and each node can only have local information (i.e., neighbor information). In particular, these challenges are reflected by the trade-offs between creating a connected scatternet and satisfying energy-aware requirements.

The ACB-tree combination occurs at the top of two trees, which brings the benefits of controlling the

tree balance and shortening its diameter, but makes the tree combination more selective than other tree formation algorithms which allow two trees to combine by connecting any two nodes [52, 73], since it is possible that two trees can only be connected by non-handle nodes. Due to the lack of global knowledge to identify the optimal procedure for growing trees, we relax the standard ACB-tree combination under certain situations (e.g., two trees can only be connected through non-handles) to ensure generating a single connected scatternet. However, since the standard ACB-tree combination obviously provides benefits for achieving energy-efficiency, we want to apply standard combinations as much as possible and avoid merging trees through non-handle nodes. This requires adjusting the combination procedure to facilitate future combinations besides considering energy-efficiency. Furthermore, the distances between devices in a multi-hop scenario can vary significantly as opposed to single-hop scenarios, so the tree-growing procedure should differentiate link lengths and select short links to achieve energy-efficiency.

4.4.2 EMTS

Considering the challenges of forming energy-aware scatternets in a multi-hop scenario, we let each device collect its neighbor information at the beginning, which provides necessary local information to proceed with the following scatternet formation. So we add a neighbor discovery phase as the starting phase to the single-hop formation technique. Furthermore, each tree's neighbor information should be updated as tree continues to grow, which enables them to make the right combination decision until the formation of a single connected scatternet. So we modify the piconet construction and ACB-tree growing phase to let each device or tree inform its neighbor about its status change after each combination. Based on these considerations, we design EMTS (Energy-efficient Multi-hop Tree Scatternet) formation technique to be composed of three phases: neighbor discovery, piconet formation and ACB-tree growing.

Neighbor Discovery

Since in the multi-hop scenario devices may be out of each other's transmission range, neighbor discovery is a commonly used phase [52, 54, 55, 39, 81] which enables each device to be aware of its surrounding neighbors and obtain neighbor information (e.g., neighbor ID, clock) for the use of future scatternet formation. In EMTS, the neighbor discovery phase helps guarantee the connectivity of the generated scatternet since it essentially identifies a potential connection graph for the scatternet. Furthermore, neighbor discovery

also provides EMTS with critical information, e.g., the neighbors' energy levels and distances, to achieve energy-efficiency of the generated scatternet.

In this phase, each device continuously contacts its neighbors by applying the widely-used neighbor discovery approach introduced in [65]. In this approach, each device independently alternates between the INQUIRY and INQUIRY SCAN states, and symmetrically exchanges information (e.g., address, clock and energy capability) with each other upon establishing a temporary connection that will be discarded after finishing information exchange. Furthermore, each pair of neighbors need to estimate their distance (link length) by measuring the received signal strength (a Bluetooth transceiver that supports power-control has a receiver signal strength indicator (RSSI) [1]). All devices stop discovering operations after a timer expires. Typically, most neighbors (over 90%) can be discovered within 6 seconds [13] or even shorter [30].

Piconet Formation

In this phase, independent devices are grouped into piconets. Since in a piconet the master has a higher workload than slaves, the most energy-capable devices will be assigned as masters. Further, since the number of piconets should be minimized due to the possible frequency collision among piconets, we attempt to make each piconet as full as possible. As in the single-hop formation technique, each piconet can have at most 5 slaves. Within the 2 free slots (a piconet can have at most 7 active slaves), the first is used to connect piconets in the following ACB-tree growing phase, and the other one is used to form temporary links to exchange neighbor information.

Since each device has already obtained its neighbors' address and clock information in the neighbor discovery phase, we use the PAGE and PAGE SCAN operation (more efficient than the INQUIRY and INQUIRY SCAN operation used in the single-hop technique) to establish master-slave links. For ease of description, we term a device A to be *stronger* (or *weaker*) than B if A's battery level is higher (or lower) than B. Initially, only the devices with the highest energy among their neighbors (i.e., strongest) assume the master roles and start to PAGE their neighbors to build their piconets (the closest neighbor will be paged first with the aim to shorten the link length). All other devices perform PAGE SCAN to wait to be contacted by one of its stronger neighbors. In Procedure 1, we provide the pseudo code of a device's initial behavior.

A device becomes a slave of the master that first contacts it if this master has not acquired enough slaves, and a device assumes the master role if all of its stronger neighbors have either become slaves or masters

Procedure 1 PiconetFormation()

```
if I am the strongest device among neighbors then
    role ← MASTER;
    PAGE the closest neighbor;
else
    Perform PAGE SCAN, wait to be contacted by a stronger neighbor;
end if
```

Procedure 2 CreateConnection(B)

```
if I am a MASTER then
    Attract B to become my SLAVE if B has not determined its role and I have less than 5 slaves;
    Record B's role and inform B about my role;
else
    Record B's role if it has made its role decision;
    Inform B about my role and my master's address;
end if
ContactNeighbors();
```

Procedure 3 ContactNeighbors()

```
PAGE the closest weaker neighbor C to attract slaves and inform my role;
PAGE a stronger neighbor D to inform my role;
Perform PAGE SCAN, wait to be contacted by a weaker neighbor E;
Finished the piconet formation phase, enter the next phase;
```

that have acquired enough slaves. After deciding its own role (master/slave), a device informs its neighbors about its role to enable them to make their own role decisions. So if a device A initializes a connection (by performing PAGE), it must have determined its own role and is now attempting to attract slaves (if A is a master) or informing its role decision (either master or slave). In Procedure 2, we provide the pseudo code for a device that creates a connection. After A exchange information with the other connection side, it will perform *ContactNeighbors()* procedure to inform its neighbors about its role decision. To form piconet efficiently and enable each device to obtain all of its neighbors' information, we require each device to follow a predefined order in *ContactNeighbors()* (Procedure 3): after being contacted by all stronger neighbors, a device will first contact weaker neighbors to attract slaves (if it is a master) and inform its role, then inform all stronger neighbors about its role, and finally wait to be informed by weaker slaves about their roles.

For a device A that receives a connection (by performing PAGE SCAN) from B, A will first record B's role (since B must have already determined its role), then A will become a slave if it has not determined its

Procedure 4 ReceiveConnection(B)

```
Record the role of B;
if B is a master then
  if I have not decided my role and B has less than 5 slaves then
    role ← SLAVE;
    master ← B;
  end if
end if
if I have been contacted by all stronger neighbors then
  role ← MASTER if I have not decided my role;
  ContactNeighbors();
else
  Perform PAGE SCAN, wait to be contacted by a stronger neighbor;
end if
```

role and *B* is a master with less than 5 slaves. After *A* has been contacted by all of its stronger neighbors, it assumes a master role if it has not determined its role, then *A* starts to attract its own slaves (if it is a master) and contact neighbors to inform its role decision by performing the *ContactNeighbors*() procedure. In Procedure 4 we provide the pseudo code for a node receiving a connection.

Eventually, all neighboring piconets (we call two piconets neighbors if two devices, one from each piconet, are neighbors) will learn of each other. Furthermore, all paths between neighboring piconets, i.e., the paths that connect two neighboring piconets' masters and involve at most one slave for each piconet, will be identified by their masters for later usage. We call these path *neighbor paths* hereafter. Since each piconet has a star topology, neighbor paths can be at most 3 hops long. These neighbor paths, when suitable, are potential candidates for the conceptual tree edges in growing ACB-trees by hooking up different nodes and trees. In the following, we prove that in each piconet the most energy-capable device will act as the master.

Proposition 1. *For each piconet, the master's energy level is higher than any of its slaves'.*

Proof. For a device *A*, we assume the neighbors that *A* has discovered is a neighbor set $N(A)$. Among the neighbors in $N(A)$, we further divide them as the stronger neighbors $N_s(A)$ and weaker neighbors $N_w(A)$. Assume *A* becomes a master. There are two possibilities:

- case 1. *A* has the highest energy capability among its neighbors, i.e., $N_s(A) = \phi$. Then *A*'s slaves, which must be its neighbors, will have a lower energy level than *A*.

- case 2. *A* has some higher-energy neighbors, i.e., $N_s(A) \neq \phi$. According to our algorithm, *A* can become a master only after all the devices in $N_s(A)$ have determined their roles and informed *A*. So all *A*'s slaves can only be chosen from $N_w(A)$, i.e., *A* has the highest energy level in its piconet. □

ACB-tree Growing

The objective of this phase is to connect piconets to grow energy-aware ACB-tree scatternets. As in the single-hop formation technique, we use a tree node to represent a piconet and use its master's energy to represent this node's. We regard a lone node as a special ACB-tree consisting only of a handle. Since ACB-trees' combination only occurs at the top (shown in Figure 4.3), we call two ACB-trees *neighboring trees* only if their handles are neighbors, and call an ACB-tree $orphan$ if it has no neighbor but can connect to other trees through non-handle nodes. For ease of description, we call and represent a handle's energy capability as the associated tree's.

General Tree-Growing Procedure: To efficiently create scatternets in a distributed fashion for the multi-hop scenario, we adopted a bottom-up method to grow trees by recursively applying ACB-tree combinations. Since the standard ACB-tree combination only occurs between neighboring trees, we want to grow trees in such a way that each of them can always identify neighbors till being merged (i.e., avoid becoming orphans), which brings ACB-tree's benefits for achieving energy-efficiency. Identifying the optimal tree growing process requires global information; since only local information is available, we utilize the number of neighbors (which can be maintained locally) as an important factor to control tree combinations since it directly affects the future choices for performing tree combinations and achieving energy-efficiency. In particular, trees with fewer neighbors should combine with others as quickly as possible to avoid becoming orphans, and after each combination the resulting tree should have sufficient neighbors to facilitate future combinations. Based on this policy, any tree T with the smallest neighbor count (k) in its neighborhood will have the highest priority to identify a neighbor to combine with (we will describe a little later how to choose the most suitable neighbor and the most energy-capable neighbor path for connecting neighboring trees). For ease of description, we term a tree *A* to be *smaller* than another tree *B* if *A*'s neighbor count is smaller than *B*.

Procedure 5 ACB-treeGrowing()

```
if I have neighbors then
  if My tree is the smallest tree among neighbors then
    Choose the most suitable bigger neighbor B;
    Establish the most energy-capable neighbor path to B;
  else
    Wait a smaller neighbor tree to establish a neighbor path to me;
  end if
else
  if I am a orphan then
    Identify a tree node to establish a connection to another tree;
    Exit;
  else
    I have finished tree growing, exit;
  end if
end if
```

The neighbor count heuristic significantly facilitates the growth of ACB-trees; however, it can not eliminate orphans completely due to the lack of global information. To ensure the generation of a single connected scatternet in the multi-hop scenario, by absorption of even these orphans, we further require each tree's handle to maintain neighbor information from all of its tree nodes and acquire such information from the retiring handle at each combination. This way, if an orphan tree T appears (which occurs infrequently), T can always select a suitable tree node, which can reach another tree T', to establish a connection to T', and hence ensure generating a connected tree scatternet. Note that the closer this tree node is to the handle (or higher in the tree levels), the more energy-capable it is and more suitable for connecting to other trees. In Procedure 5 we provide the pseudo code of devices' initial behavior in this phase (note that all of the following code is only performed by the master of the handle node in each tree, i.e., the leader).

The whole ACB-tree growing procedure is composed of a set of rounds. In each round, each tree will wait until all its smaller neighboring trees finish the combination, and then start to choose a bigger neighbor to combine with if all its smaller neighbors do not combine with it in the current round. After combination (note that it is possible for a tree not to combine with another during a round if all its neighbors have been combined with others), it should inform its neighbors about its combination status to enable the tree-growth in the next round. Both tree combination and status informing require the establishment of neighbor path connections (note that we leave a free slot in each piconet to establish these neighbor paths). A neighbor

Procedure 6 NeighborTreeConnection(B)

```
if I have not combined in the current round then
   Perform ACB-tree combination with B if B has not combined in the current round;
else
   Inform B about my current combination status;
   Record B's current combination status;
end if
Wait to be contacted by all smaller neighbor trees;
if I have not combined in the current round then
   Create the most energy-capable neighbor path to the most suitable bigger neighbor;
else
   Inform all smaller neighbors about my current combination status;
   Wait to be informed by all bigger neighbor trees about their current combination status;
end if
if My handle is selected to be the new handle of the combined tree then
   Enter the next round and perform initial behavior as in Procedure 5;
else
   Exit;
end if
```

path connection used for informing purposes is temporary and will be disconnected afterwards. Once a tree A successfully creates a neighbor path to a neighbor B, we can tell if A wants to combine with B or inform B based on A's combination status in the current round. After tree A finishes its combination and learns all its neighbors' combination status, and if A's handle is selected to be the new handle of the combined tree (new handle selection is based on energy level and neighbor count, as described later), A will enter the next round and perform the initial behavior (Procedure 5) iteratively until a single, connected tree scatternet has been formed. For a new-combined tree, its handle node will have the global knowledge of this tree by exchanging the two original tree's information at the top. In Procedure 6, we provide the pseudo code for the tree that connect to a neighbor B.

Neighbor Tree and Path Selection: Compared to other scatternet formation algorithms, EMTS lets each tree incur overhead to maintain neighbor information. However, this information allows EMTS to not only combine ACB-trees distributely and control tree balance, but also achieve other energy-efficient requirements, e.g, short link length and assigning nodes with higher energy at higher tree levels. In particular, all links and bridges in a scatternet are determined by the neighbor paths (or tree edges, in the ACB-tree)

selected for tree combinations, so based on the neighbor information we can identify energy-capable neighbor paths that have short links, energy-capable bridge devices, and fewer hops for each tree combination. Assume that a tree T's neighbor-tree set is $N = \{T_1, \cdots, T_k\}$: for each neighbor T_i ($1 \leq i \leq k$), T also identified a set of neighbor paths $NP_i = \{p_1, \cdots, p_m\}$, where each neighbor path p_j ($1 \leq j \leq m$) can contain at most 4 devices (i.e., $|p_j| \leq 4$). We represent $p_j = \{d_1, \cdots, d_l\}$, where d_x ($1 \leq x \leq l \leq 4$) is a device along this path. To reflect the neighbor paths' energy-capabilities, we define an EC metric for a path p_j:

$$EC(p_j) = min_{d_x}\{\frac{BATT(d_x)}{max\{|d_{x-1}d_x|, |d_x d_{x+1}|\}}\}/|p_j| \tag{4.1}$$

where $d_x \in p_j$ and $BATT(d_x)$ denotes d_x's battery level, $|d_{x-1}d_x|$ and $|d_x d_{x+1}|$ denote the physical length of the two links from/to d_x in the p_j. EC calculation accounts for the link length, bridge devices' energy levels, and number of hops in the neighbor path. So a path with higher EC will be more energy-capable and suitable for tree combinations.

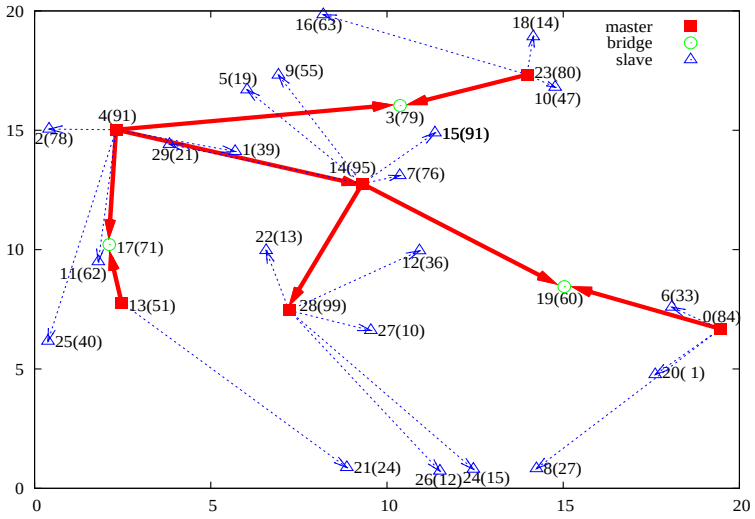
Based on EC metric for neighbor paths, we further define a *SUIT* metric for a neighbor-tree T_i to represent its suitability to combine with T:

$$SUIT(T_i) = max_{p_j} EC(p_j) - \alpha|h(T_i) - h(T)| \tag{4.2}$$

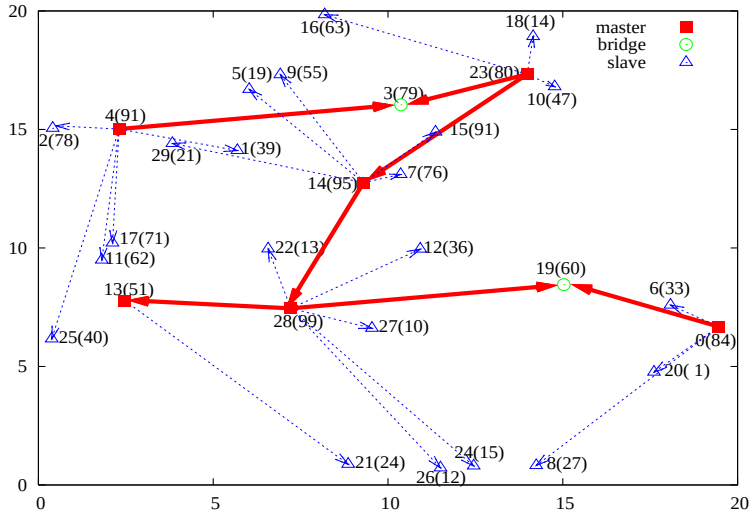
where neighbor path $p_j \in NP_i$ of neighbor tree T_i, $h(T_i)$ denotes T_i's height, and α is a normalizing factor. So besides considering the energy-capability of neighbor paths, the $SUIT$ metric also accounts for the height difference among neighboring trees which helps to control the tree balance. A tree T will identify the neighbor tree T_i with the highest $SUIT$ and use the path with the highest EC for combination. Note that all the information needed for the calculation of EC and $SUIT$ can be maintained locally. After a tree combination, the handle with more neighbors is preferred to be the new handle to facilitate future combinations; on the other hand, the new handle should also be more energy-capable since it will appear at higher tree levels (as in the single-hop scenario). So we compare the two original handles' neighbor-counts and energy capabilities, and choose the more energy-capable one if their energy capabilities vary too much (exceed a threshold); otherwise we choose the one with more neighbors as the new handle.

An Example: To recap this scatternet formation process, in Figure 4.5(a), we illustrated a generated ACB-tree with 30 devices in a $20m \times 20m$ area. For clarity, we show an arrow from a master to a slave in the piconet: we use thick red arrows to denote neighbor paths (i.e., inter-piconnet connections), and use dotted blue arrows to denote intro-piconet connections. The numbers in the parenthesis represent the devices' energy levels, and only the S/S bridges are distinctly identified as bridges, while M/S bridges are regarded as masters. The scatternet includes six piconets/nodes, whose masters are 4, 13, 28, 14, 23, and 0 (from left to right). We can observe that piconets group close-by devices and their masters have higher energy levels than slaves. This ACB-tree scatternet grew in the following manner: initially node 0, which has the fewest number of neighbors since it is located close to the right-edge of the area, starts to identify a neighbor to combine with. Note that it is important to let node 0 combine first since it is more likely to become an orphan if its neighbors combine with others first and no longer act as handles. According to the $SUIT$ metric, node 0 combines with node 14 into a 2-node tree $T_{14,0}$ through the neighbor path $p_{14,0} = \{14, 19, 0\}$ that has the highest *EC*, and node 14 is selected as the new handle since it has more neighbors and a higher energy level. Similarly, the remaining nodes form two two-node trees $T_{4,23}$ (through the neighbor path $p_{4,3,23}$) and $T_{28,13}$ (the neighbor path between 28 and 13 is not shown in the final scatternet since it will be torn-down when performing the succeeding tree combinations), and nodes 4 and 28 are selected as the new handles for these two trees. Following a similar procedure, $T_{4,23}$ and $T_{28,13}$ combine into a new 4-node tree $T_{28,4,13,23}$ with node 28 selected as the new handle since it has the higher energy level (both nodes 28 and 4 have one neighbor tree $T_{14,0}$). Finally, the remaining two trees $T_{28,4,13,23}$ and $T_{14,0}$ combine into the final scatternet, and node 28 will act as the handle due to its higher energy level. Following the tree from the handle 28, we can see that nodes at higher tree levels have higher energy levels, and the tree edges or the neighbor paths (thick lines) typically involve shorter links, fewer hops, and more energy-capable bridge devices. Further, the generated tree is well-balanced with a short diameter and a maximum device degree of 2 (each device belongs to at most 2 piconets) since ACB-tree is binary and grows in a balanced fashion.

For comparison, in Figure 4.5(b) we provide an optimal scatternet that achieves the maximum lifetime (defined as the time that the first device depletes its energy) when all masters (or M/S bridges) send packets to all others periodically. For simplicity, this optimal scatternet is generated through a brute-force searching based on the piconets created by EMTS, which provides a lifetime close to the real optimal scatternet



(a) Scatternet formed by EMTS



(b) Optimal Scatternet with the Maximum Lifetime

Figure 4.5: Comparison between ACB-Tree Scatternet and Optimal Tree Scatternet

but allows the scatternet formation with an acceptable computation complexity (it took about 70 hours to calculate this optimal result on a Pentium 4 computer). The difference between these two scatternets is that in EMTS node 14, the root of the tree, is assigned the highest workload (node 28 has the highest energy level and is assigned as the handle to facilitate possible future tree combinations), while in the optimal scatternet node 28 handles the highest workload. Though this makes EMTS less energy-efficient than the optimal one, it provides EMTS with an additional energy-capable node (the handle) that has global knowledge and

can handle additional tasks for the whole scatternet. On the other hand, the two tree-scatternets do show similar features of assigning roles to nodes that suit their workload and energy level, e.g., by following the handle/root node 28 in the two trees most nodes at higher levels have higher energy levels than their children. Further, these two trees also show similarities on selecting paths to connect nodes, e.g., the paths $\{4, 3, 23\}$ and $\{14, 28\}$, etc, which are more "energy-capable" than the alternatives. Note that the lifetime definition depends on the most energy-weak device in the network, while not consider the overall energy consumption, so the optimal scatternet, though can achieve longer network lifetime, has a longer diameter than EMTS and has a higher overall energy consumption.

4.4.3 Scatternet Maintenance

We propose a local reorganization technique to maintain energy-efficiency and extend the network lifetime in a multi-hop network. Further, since bluetooth devices are usually deployed in dynamic environments, EMTS is designed to handle device joins and departures in an energy-efficient manner.

Local Reorganization

Depending on the application requirements, devices in the scatternet usually have different workloads, which makes some devices (usually masters and bridges) drain their energy more quickly than others. To avoid partitioning scatternet, it is desirable to alleviate the workload of the devices that are going to deplete their energy and hence extend the network lifetime. Reorganization is a widely-used technique to maintain certain properties of the scatternet [52][22], and in EMTS we propose a reorganization technique to adjust the devices' workload in order to extend scatternet lifetime. Since reconfiguration inevitably affects normal communication among devices, a light-weight local reconfiguration is preferable than global ones. Further, the multi-hop nature of the scatternet restricts reorganization choices since new links can only be created between neighboring devices. For EMTS, we designed a local reorganization technique that has minimal changes in the number of links to reduce reorganization overhead.

The idea behind our reorganization technique is to relieve the workload of the energy-depleting devices. Such devices could be either pure slaves, S/S bridges, or masters (including M/S bridges). Pure slaves already have the lowest workload, and their failing does not affect the rest of the scatternet. The S/S bridge bottlenecks can be easily handled by identifying an alternative path to connect the two nodes that the original

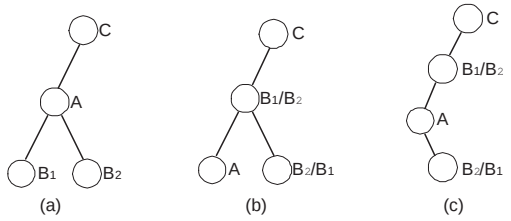


Figure 4.6: Reconfiguration Topologies

bridge connects. In the following we focus on master bottlenecks. For ease of description, we do not differentiate between a master and its associated node (or piconet) since we do not modify the intra-piconet links to reduce the reconfiguration overhead.

Considering that the tree-level directly affects a device's workload (i.e., higher the tree level, higher the workload), we design our reorganization as a sinking operation to move an energy-weak node one level down to relieve its workload. In an ACB-tree, the topology formed by EMTS, a parent's workload is twice its child's since there are double the number of devices in the parent's subtree. So this one-level sinking operation, though light-weight, can effectively reduce the affected node's workload and extend scatternet lifetime. On the other hand, at a time, sinking a node deeper in a multi-hop scenario increases the reorganization overhead considerably or even may not be possible.

Our reorganization will be triggered when a node *A*'s energy level drops below one of a few pre-defined thresholds. Figure 4.6(a) shows such a node A with parent C and two child nodes B_1 and B_2. To sink *A*, one of its children will be promoted one level up to replace *A*, i.e., forming new links with *A*'s parent *C* and possibly *A*'s other children. Fortunately, this can be performed by using the efficient PAGE and PAGE SCAN operation since these nodes can maintain each other's clock information. Depending on *A*'s children and their energy-levels, there usually exist multiple choices to perform the sinking operation. In Figure 4.6 (b) and (c), we illustrate the reorganization choices (note that node B_1 and B_2 are interchangeable).

To select the topology that leads to the best lifetime extension, we calculate a weight ω for each reorganization topology (RT) that reflects the scatternet lifetime estimation with this reconfiguration, and then choose the one with the highest ω that represents the longest lifetime extension. Note that our lifetime estimation is restricted to the four nodes shown in Figure 4.6 to make our reorganization local and reduce the overhead. Lifetime estimation usually requires the network traffic information, which cannot be predicted ahead since

the network traffic is usually application-dependent. We assume an all-to-all traffic for lifetime estimation since it involves all devices without bias. Assume that a reorganization topology RT includes a set of devices $D(RT) = \{d_1, d_2, \cdots, d_n\}$, we calculate an all-to-all route set $R(RT) = \{r_{1,2}, r_{1,3}, \cdots, r_{n-1,n}\}$, where $r_{i,j} = \{d_i d_{i+1}, \cdots, d_{i+k} d_j\}$ represents the route from d_i to d_j, and $d_l d_{l+1} (i \leq l < j)$ denotes a link in $r_{i,j}$. For each route, any two neighboring nodes will use the neighbor path with the highest EC value, i.e., the most energy-capable path. We assume the energy for packet transmission (P_t) is proportional to the square of link length, i.e., $P_t(d_i d_j) = \rho |d_i d_j|^2$, where ρ is a constant, and further assume the energy spent for receiving a packet (P_r) is a constant value, i.e., $P_r(d_k d_i) = \nu$ [52]. So in a RT the energy spent by a device d_i under all-to-all traffic can be estimated as:

$$\begin{aligned} E(d_i) &= \textstyle\sum_{d_j} P_t(d_i d_j) + \sum_{d_k} P_r(d_k d_i) \\ &= \textstyle\sum_{d_j} \rho |d_i d_j|^2 + \sum_{d_k} \nu \end{aligned} \tag{4.3}$$

where $d_i d_j \in r_m \in R(RT), d_k d_i \in r_n \in R(RT)$. It gives us RT's lifetime (i.e., its weight ω):

$$\omega(RT) = min_{d_i} \{ \frac{BATT(d_i)}{E(d_i)} \} \tag{4.4}$$

where $d_i \in D(RT)$. Since ω essentially reflects the expected lifetime of a local network, we calculate ω for all the possible reconfiguration topologies and the original topology (Figure 4.6). The reconfiguration with the highest ω will be performed accordingly if it achieves a higher ω than the original topology.

Device Arrival

Since leaf nodes typically have low routing overhead and our experiments showed that the leaf nodes' dedicated slaves cover the majority of the scatternet area (over 95% area for 50 devices in a $30m \times 30m$ area), we let each leaf node's dedicated slaves (non-bridge slaves) spend a small portion of time to discover new devices by periodically performing INQUIRY SCAN. The remaining devices will also perform INQUIRY SCAN occasionally to guarantee the absorption of any new devices that will perform INQUIRY to access the existing scatternet. This way, we also save energy besides discovering and absorbing new devices efficiently.

Device Departure

The trivial case is when a dedicated slave departs and does not affect the scatternet. The complicated case occurs when a bridge/master B departs, which may break the connection among two or more nodes. Without loss of generality, we assume B's departure disconnects a parent node P from its child node C. P and C will establish another path connection if they are still neighbors. Otherwise, C will ask its sub-branch devices to perform INQUIRY such that its branch will eventually be reconnected to the scatternet. It should be noted that our local reorganization procedure can be used to maintain network's energy-efficiency after changing network topologies, i.e., absorbing new devices or healing a scatternet.

4.5 Performance Evaluation

We have implemented EMTS on Blueware 1.0 [2] which is built on top of the network simulator ns-2 [5] and closely follows Bluetooth specification 1.1 [1]. We compare EMTS with three other tree-based scatternet formation algorithms: TSF [73], SF-DeviL [52] and SHAPER [22] and a mesh-like algorithm BlueMesh [55]. Both single-hop scenarios ($7.07m \times 7.07m$) and multi-hop scenarios ($30m \times 30m$) are utilized for comparison: TSF is a single-hop algorithm, and SF-DeviL only report results in the single-hop scenario [52]; SHAPER and BlueMesh will be used for comparison in the multi-hop scenerio. TSF and SHAPER have a readily-available implementation on Blueware, so the data reported for them are obtained by replicating the experiments. The results for SF-DeviL and BlueMesh are directly taken from [52] and [55], respectively. Each data point reported is the average result of 30 simulation runs with varying seeds. Considering that the device discovery phase is widely used in multi-hop scatternet formation algorithms and most neighbors (over 90%) can be discovered within 6 seconds [13] or even shorter [30], we assume about 90% percent neighbors can be discovered in the neighbor discover phase (though it is not strictly required for EMTS).

We conducted several simulations to evaluate both scatternet formation and maintenance. For scatternet formation, we provide results about scatternet lifetime without reorganization, average link length, scatternet diameter, number of piconets, formation delay and message complexity. For these results, all devices start the scatternet formation process at the same time, though this is not a strict requirement for our algorithms. For scatternet maintenance, we studied the scatternet lifetime with reorganization, reorganization delay and new device absorption delay.

4.5.1 Scatternet Formation

To identify the energy-efficiency of scatternets, we conducted the lifetime evaluation for EMTS. Similar to SF-DeviL, we define lifetime as the duration until one of the devices exhausts its energy, and assume each device has power control and a receiver sensitivity of -60dbm, the receiving power P_r is the same as the maximum transmission power P_t, and the energy spent in standby mode is ignored [52]. However, SF-DeviL's initial device energy assignment and traffic flow pattern are unclear. We assign each device d a random initial battery level $E(d)$ mWH ($1 \leq E(d) \leq 100$) and randomly generate $n/2$ (n is the number of devices in the scatternet) traffic flows of 40 packets/s. This traffic pattern implies that "on an average, each node is involved in one traffic connection either as source or destination" [29]. We use the free space power loss model $P = \rho P_t / l^2$ [62], where P_t and P denote the transmission power and received power, respectively, where l is the link length and ρ is a constant factor. For the sake of comparison, we added necessary code to TSF and SHAPER for lifetime measurement, and provide the results in Figure 4.7. Since EMTS deliberately shortens scatternet's average link length and diameter, and assigns devices' roles based on their energy capabilities, it substantially increases TSF's lifetime by 225% to 937% in the single-hop scenario, and increases SHAPER's lifetime by 175% to 719% in the multi-hop scenario. As the network density increases, though all algorithms' lifetimes decrease due to the increase of traffic flows (except SF-Devil that may use a different traffic pattern), EMTS decreases slower than others, since a denser network provides EMTS with more opportunities to optimize the energy-aware topology besides ensuring scatternet connectivity (that is also why EMTS, as expected, has a longer lifetime in the single-hop scenario than in the multi-hop scenario). We also include SF-DeviL's results directly taken from [52] (SF-DeviL reports up to 50 devices). However, since SF-DeviL may use a different traffic flow pattern and energy assignment model, comparison is useful only in a qualitative sense.

In Figure 4.8, we present the simulation results of the number of piconets (both single-hop and multi-hop scenario results are provided). Since EMTS algorithm includes a separate piconet formation phase in which each piconet is allowed to have at most 5 dedicated slaves, EMTS has the fewest number of piconets of all the presented algorithms. In Figure 4.9, we provide the average device degree. It can be observed that EMTS generates the scatternets with smaller average device degree than other algorithms, since their tree-growing phase attempts to build a binary tree.

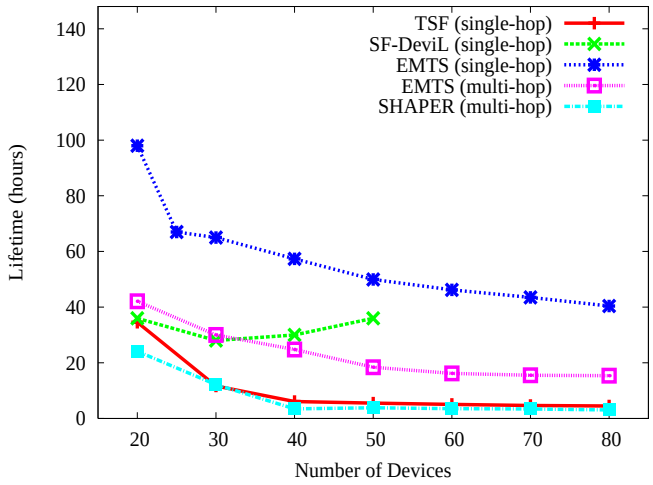


Figure 4.7: Scatternet Lifetime

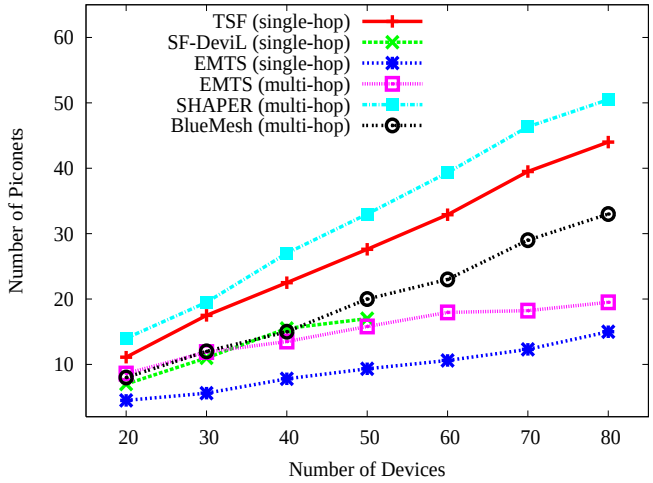


Figure 4.8: Number of Piconets

In Figure 4.10, we show the average link length (the single-hop and multi-hop scenario should be compared separately). Though in the multi-hop scenario EMTS naturally has longer link length than that in the single-hop scenario, it outperforms all other tree-based scatternet formation algorithms in both scenarios, because, in EMTS, each combination occurs between physically close trees and the path for combination is carefully selected to shorten the link length. Furthermore, the link length of EMTS decreases (as expected) as the network becomes denser. We show scatternet diameter in Figure 4.11. It can be observed that EMTS forms the scatternets with the shortest diameter in both single-hop and multi-hop scenarios (BlueMesh [55]

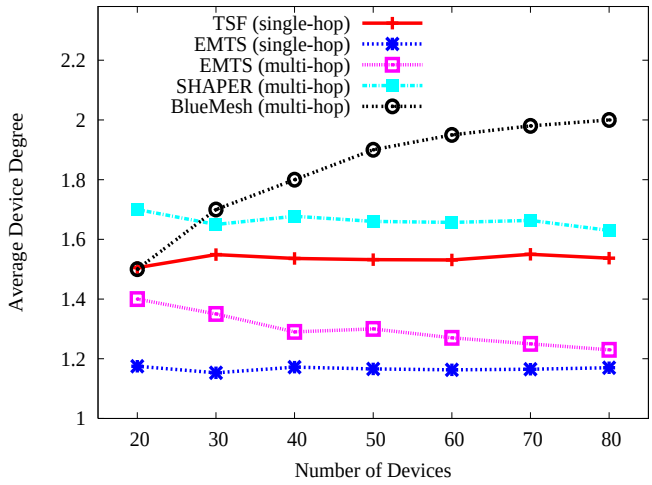


Figure 4.9: Average Device Degree

only reported the average path length), since its controlled ACB-tree combination process improves the tree balance, which leads to shorter diameter.

In Figure 4.12, we provide the formation delay comparison results. It should be noted that we do not count the neighbor discovery delay for EMTS and SF-DeviL. We can observe that EMTS's formation delay is shorter than SF-Devil; compared to others, it increases quickly as the network becomes denser due to its additional communications between neighboring piconets/trees to identify the best choice during scatternet formation. However, EMTS can still be formed within a reasonable time period especially in the multi-hop network, since most communication links are created by performing the efficient PAGE operation. In Figure 4.13 we provide the message complexity which is proportional to the number of links established. EMTS has higher message overhead than TSF and SHAPER since each tree/node needs to communicate with all its neighbors after each combination to maintain neighbor information and form energy-aware scatternets.

4.5.2 Scatternet Maintenance

Reorganization is an effective technique for maintaining an energy-efficient scatternet in order to extend scatternet lifetime. In Figure 4.14, we show the lifetime extension results when reorganization is applied. It can be observed that EMTS's scatternet lifetime is increased by 31-112% in the multi-hop scenario and 33-53% in the single-hop scenario. The effect of reorganization is more pronounced in the multi-hop scenario especially when the network becomes denser, because the average communication distance (link length) is

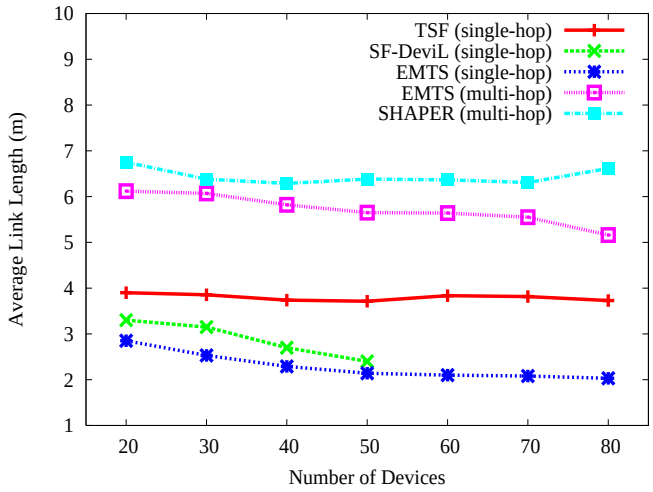


Figure 4.10: Average Link Length

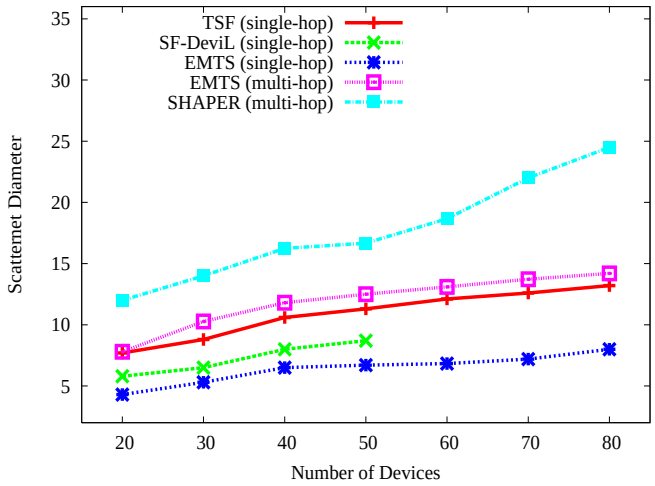


Figure 4.11: Scatternet Diameter

much longer in the multi-hop scenario (as shown in Figure 4.10), which enlarges the difference of energy consumption for the nodes located at different tree-levels.

Though being an effective approach to extend scatternet lifetime, reorganization would inevitably interrupt the communication of the involved devices, so reorganization delay as well as the number of devices involved in the reorganization should be minimized. In Figure 4.15, we provide the reorganization delay as a function of network size for EMTS and SHAPER. Clearly EMTS's reorganization delays is significantly shorter than that of SHAPER, because instead of performing a global reorganization which requires the

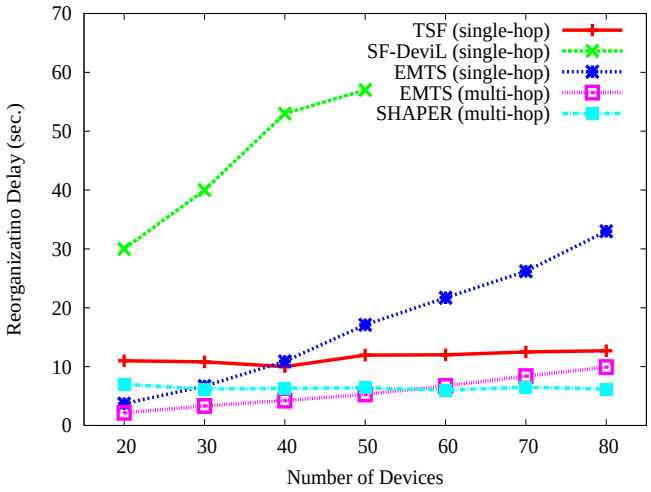


Figure 4.12: Formation Delay

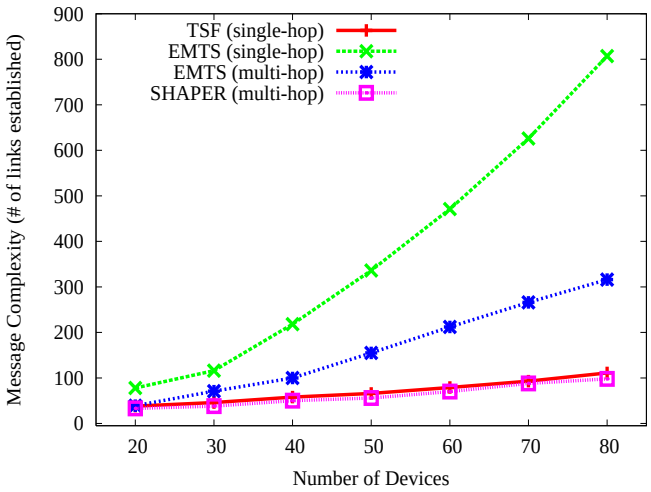


Figure 4.13: Message Complexity

participation of all devices as in SHAPER, EMTS executes a localized reorganization that is restricted to a small local tree area, which reduces the reorganization overhead to a constant.

To evaluate the efficiency of new arrival absorption approaches in EMTS, we replicated the simulation in SHAPER [22], i.e., 25% of the devices on x-axis arrives en masse after a scatternet has been formed with the rest of devices, and compared the period of time spent to accommodate all new devices among EMTS, TSF and SHPAER in Figure 4.16. It can be observed that EMTS can absorb new devices efficiently, and its absorption delay decreases as network becomes denser since more devices will be involved for new arrival

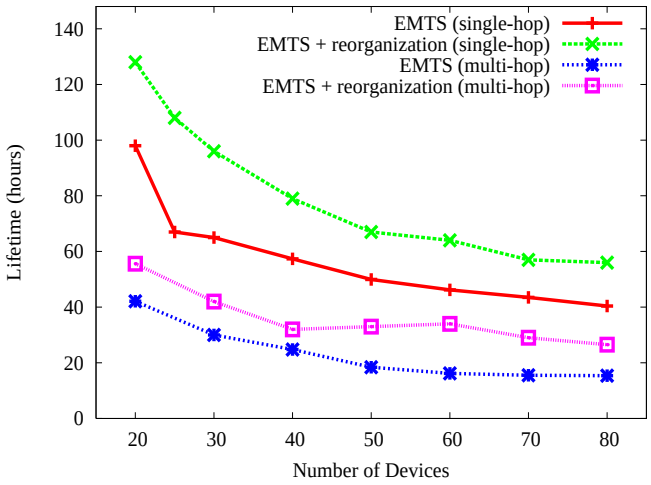


Figure 4.14: Lifetime Improvement with Reorganization

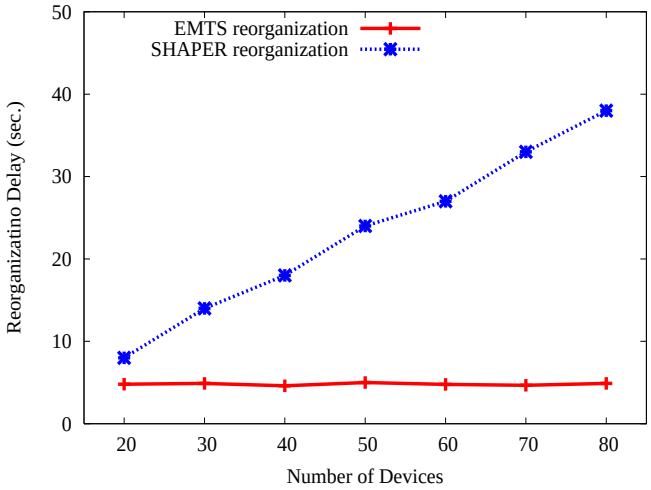


Figure 4.15: Reorganization Delay Comparison

discovery.

4.6 Summary

We proposed a distributed algorithm to create energy-aware multi-hop tree-based scatternets (EMTS). EMTS is designed based on an ACB-tree topology, which is a complete binary tree with an additional node connected to the root. ACB-tree's distinct features facilitate to grow trees in a balanced manner and reduce control tree's height. ACB-tree's binary nature also help to minimize node degree.

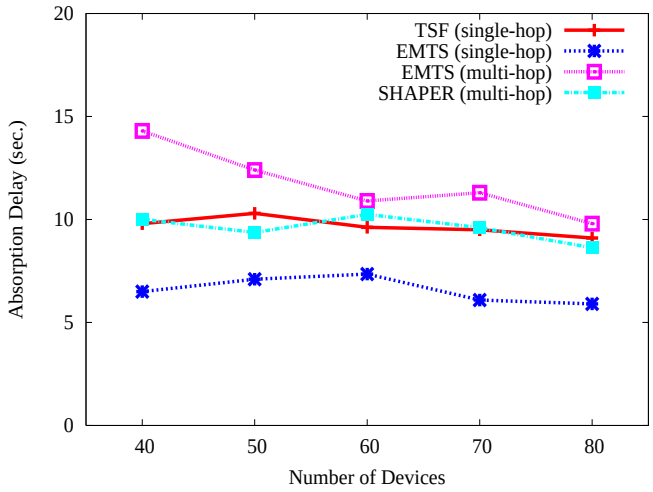


Figure 4.16: New Device Absorption Delay

EMTS forms energy-efficient scatternets by recursively applying ACB-tree combinations that are carefully controlled to improve tree balance and shorten tree links. Further, EMTS assigned devices' roles based on their energy capabilities such that higher workloads are handled by more capable devices. These features enable EMTS to have significantly longer lifetimes than others (by up to 7-9 times), and also achieve the desired characteristics of short link length, small diameter, low device degree and low number of piconets. The price that EMTS affords to achieve energy-efficiency is the relatively higher formation delay and communication overhead. To operate energy-efficient scatternets, we addressed EMTS's maintenance by proposing local reorganization techniques that further extend scatternet lifetime with low overhead, and handle device arrival and departure efficiently.

CHAPTER FIVE

CONCLUSIONS

5.1 Summary

In this dissertation, we presented several energy-efficient topologies and protocols for wireless sensor networks (WSN) and personal area networks (PAN). In general, the devices in these type of networks are battery-powered and resource-constrained, so energy-efficiency has been regarded as the primary designing objective for these two networks. On the other hand, applications in WSN and PAN also have performance requirements in terms of end-to-end delay, throughput, etc. So the focus of this dissertation is to improve network energy-efficiency (usually reflected by the network lifetime) without sacrificing performance.

For wireless sensor networks, we proposed an energy-efficient contention resilient MAC (ECR-MAC) protocol. Most existing MAC protocols in WSN either achieve energy-efficiency by sacrificing performance (e.g., [88, 42, 67]), or introducing additional synchronization or hardware overhead to resolve the tradeoff between energy-efficiency and network performance, e.g., [42, 23]. Furthermore, most of them ignored the spatially-correlated contention prevalent in sensor applications. ECR-MAC is designed based on a Dynamic Forwarder Selection (DFS) mechanism, in which each node can identify multiple potential forwarders and dynamically choose the suitable real forwarder according to the current traffic condition. DFS enables ECR-MAC to improve both network energy-efficiency and performance (e.g., end-to-end delay) without introducing synchronization or hardware overhead, since allowing each node to maintain multiple forwarders, which choose their wakeup times independently, significantly reduces the setup latency. ECR-MAC further addresses the spatially-correlated contention by diffusing traffic from different sources in a common event detection place and dispersing the time for sending reports, and hence improves network throughput and minimizes energy wastage under contention. We evaluated ECR-MAC by using both ns2 simulator and MICA2 mote testbed. Both results confirmed that ECR-MAC achieves better energy-efficiency, delay, and throughput under various traffic conditions.

Besides reducing energy consumption of each sensor node, we further improved ECR-MAC's energy-efficiency by proposing an energy balance technique. We studied the energy hole problem prevalent in WSN

that is caused by the convergecast traffic pattern, and analyzed the energy consumption caused by both idle-listening and traffics. We then proposed a differential duty cycle assigning approach that adjusts nodes' duty cycle according to their distances to the base station, which effectively compensates the skewed energy consumption caused by convergecast. Our differential approach can be generally applied to most WSN MAC protocols that use duty cycles, as long as they are not critically depending on uniform duty cycles for all nodes. We performed analysis and simulation experiments to evaluate our differential approach, and both showed that it can generally improve network lifetime over 20% depending on the network environment.

Though ECR-MAC improves energy-efficiency and network performance in general, its worst-case end-to-end delay can still be large due to its CSMA-based nature. To satisfy the requirements of certain real-time sensor application where a short bounded end-to-end delay is desirable, we explored approaches that leverage additional information (e.g., synchronization, location) to reduce the delay bound without sacrificing energy-efficiency. We studied several representative wakeup scheduling approaches (e.g., staggered and multi-parent wakeup scheduling) that coordinates nodes' wakeup times to facilitate packet deliveries, and proposed a novel sleep-based topology control technique to implement a combined wakeup scheduling that achieves shorter delay bound, better energy-efficiency, and higher throughput. Our topology control organizes the whole network as a set of rings centered at the base station, and carefully choose nodes' forwarders and coordinated their wakeup times such that each node can identify multiple forwarders whose wakeup times are evenly distributed. We further staggered the wakeup times of the nodes at neighboring rings to combine multi-parent and staggered wakeup scheduling. We evaluated our topology control and the corresponding wakeup scheduling approach by providing analysis and simulation results. Both showed that our approach, which allows to use low duty cycles, can provide the end-to-end delay that is bounded by a small factor (about 2.45) to a fully-active approach, achieve comparable throughput, and conserve energy by an order of magnitude.

In the last study, we explored the energy-efficiency issue in Bluetooth-based personal area networks. An energy-efficient Bluetooth scatternet should have small links that reduce transmission power, small diameter that reduces the overall energy consumption, and assign devices' roles according to their battery levels and traffic load. To facilitate constructing energy-aware Bluetooth scatternets, we designed an ACB-tree data structure, in which an additional node, termed handle, is connected to the root of a complete binary

tree. ACB-trees can be grown efficiently by applying tree-combinations recursively, and the generated trees have the desirable features such as small diameter, low node degree, and assign devices' roles according to battery levels. Based on ACB-tree, we proposed a Energy-aware Multi-hop Tree Scatternet (EMTS) formation algorithm, which can construct energy-efficient scatternets in the network where two Bluetooth devices are not required to be within each other's transmission range. In EMTS, we carefully control the tree-combination procedure such that small links are used for tree construction, and the devices with higher battery levels will be located at higher tree levels to balance devices' energy consumption. We also addressed the scatternet maintenance issue to keep the energy-efficiency when operating scatternets, and proposed a low-cost local reorganization technique that adjusts scatternet topology according to the changes of devices' battery levels. Our maintenance technique can also handle dynamic device arrivals and departures. We evaluated EMTS and our scatternet maintenance technique by using Blueware simulator [2]. Results showed that EMTS can significantly extend network lifetime compared to other scatternet formation techniques (by up to 7-9 times); our scatternet maintenance technique can further extend scatternet lifetime with low overhead, and handle device arrival and departure efficiently.

5.2 Future Research Directions

In the future research, we will explore to extend the techniques used in ECR-MAC, topology control, and PAN scatternet construction. In the following we provide four most related topics.

5.2.1 Local Coordination in MAC protocols

The ECR-MAC that we proposed is a pure CSMA based MAC protocol, in which each node attempts to access the channel as soon as its reports are ready. This design simplifies MAC protocol, but usually leads to the "busy wakeup" phenomenon, i.e., a node keeps sending wakeup messages to wake up its forwarders, which causes energy and bandwidth wastage since a lot of wakeup messages are discarded. Busy wakeup problem can also affect the result of applying our differential duty cycle assigning approach, since a node needs to spend more energy for waking up forwarders with lower duty cycles.

Busy wakeup is caused since nodes do not have their forwarders' duty cycle information (ECR-MAC does not use synchronization). A possible improvement over ECR-MAC is to let each node exchange local information with its neighbors such that a node can be aware of its forwarders' wakeup time. It

should be noted that this local information exchange incurs much lower overhead than the network-wide synchronization deployed in other MAC protocols, and it can effectively resolve the busy wakeup problem since each node only needs to send one wakeup message when one of its forwarders wakes up. More importantly, this local wakeup time information exchange can help to coordinate neighboring nodes' access to the channel and hence further resolve the spatially-correlated contention. For instance, each node can access the channel based on the estimation of its neighbors' access behavior, which improves the network performance (e.g., throughput) and conserves energy. So how to make use of neighbor information and perform local coordination is a research issue that deserves further study. A possible coordination approach is to set up a local order to an event, i.e., the nodes that detect the event determines its priority to send reports based on certain information that can be obtained and calculated locally. Such kind of information can be the distance to the event-happening position (it requires each node to be aware of its location) and its neighbors' wakeup times.

5.2.2 Dynamic Duty Cycle Assignment

Together with ECR-MAC, we proposed a differential duty cycle assigning approach, in which each node determines its duty cycles based on its distance to the base station. Since sensor nodes are usually expected to be static throughout their lifetime, nodes' duty cycles are determined at the beginning and do not change over time (we call it static duty cycle assignment). However, since the traffics in WSN are not necessarily uniform, static duty cycle assignment can cause unbalanced energy consumption among nodes that have similar distance from the base station (i.e., in the same corona). In our experiments, we observed that the nodes' energy consumption in the same corona can vary by over 50%.

A possible approach to resolve this problem is to incorporate dynamic duty cycle assignment, i.e., adjust nodes' duty cycles according to their residual energy. Dynamic assignment can further improve the energy balance over the whole network, however, it can also adversely affect the network performance as nodes' battery levels drop (they will reduce their duty cycles). So a possible research direction is to combine static and dynamic duty cycle assigning approach, and study their influence to the network performance in an integrated manner.

5.2.3 Incorporating Data Aggregation in Sleep-based Protocols

For typical sensor applications, nodes collaborate for event detection and reporting. So the data collected by sensor nodes that detect the same event are usually strongly correlated. To save bandwidth as well as energy consumption, it is desirable to combine the data in the network and prevent redundant information reaching the base station. In the literature, intensive work have been conducted to perform in-network data aggregation/processing [33, 10, 21]. However, none of them considered duty cycles' influence on data aggregation. A natural effect of applying duty cycles is that the network topology changes over time, which challenges the traditional data aggregation methods which are usually designed based on fixed network topologies. Furthermore, the tree topology is by nature the most suitable topology for data aggregation, since each tree node can be regarded as a converge point of all its subtree nodes. While as we have pointed out in ECR-MAC, tree topology is inherently prone to failures and lead to intensive spatially-correlated contention in WSN especially when duty cycles are applied. It is desirable to integrate topology control, wakeup scheduling, and in-network data aggregation to improve network throughput, energy-efficiency, and ensure a short end-to-end delay.

5.2.4 Applying ACB-trees in WSN

We have shown that ACB-tree can be used to construct energy-efficient topologies efficiently, and we have used it to form energy-aware bluetooth-based PAN. Similar to PAN, WSN can also employ ACB-tree to organize its topology with the aim to achieve energy-efficiency. For instance, one of the major benefits of ACB-tree is that it assigns nodes' roles to suit their battery level. Since sensor nodes may consume energy with various rates, ACB-tree can effectively extend WSN lifetime by relieving the workload of energy-weak nodes. ACB-trees also provide the advantages of short diameter and low node degree. However, applying tree topologies in WSN should coincide with the protocols that use duty cycles, since tree topology can impair network throughput and intensify spatially-correlated contention without careful consideration. So directly applying ACB-tree in WSN may achieve energy-efficiency with the price of performance degradation. A future research direction is to explore the possibility of using ACB-trees in WSN to improve both energy-efficiency and performance.

PUBLICATION FROM DISSERTATION

- M. Medidi and Y. Zhou, “Extending Lifetime with Differential Duty Cycles in Wireless Sensor Networks”, accepted by IEEE GlobeCom, November, 2007.
- Y. Zhou and M. Medidi, “Sleep-based Topology Control for Wakeup Scheduling in Wireless Sensor Networks”, in IEEE SECON, June, 2007.
- Y. Zhou and M. Medidi, “Energy-efficient Contention-Resilient Medium Access for Wireless Sensor Networks”, in IEEE ICC, June, 2007.
- Y. Zhou and M. Medidi, “An Energy-aware Multi-hop Tree Bluetooth Network”, in IEEE ICC, June, 2007.
- M. Medidi and Y. Zhou, “Maintaining an Energy-efficient Bluetooth Scatternet”, in IEEE IPCCC, April, 2006.
- M. Medidi, J. Campbell, Y. Zhou and S. Medidi, “Bounded diameter tree scatternets for Bluetooth WPANs”, in Defense and Security Conference on Digital Wireless Communication, 2005.

BIBLIOGRAPHY

[1] "The bluetooth special interest group," http://www.bluetooth.com.

[2] "Blueware simulator," http://nms.lcs.mit.edu/software/blueware.

[3] "Dmac," http://ceng.usc.edu/∼anrg/downloads.html.

[4] "Mote," *http://www.xbow.com*.

[5] "Ns-2 network simulator," http://www.isi.edu/nsnam/ns.

[6] "Power aware sensing tracking and analysis," in *http://pasta.east.isi.edu*.

[7] "Ieee standard 802.11," in *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999.

[8] O. B. Akan and I. F. Akyildiz, "Event-to-sink reliable transport in wireless sensor networks," in *IEEE/ACM Transactions on Networking*, vol. 13, 2005, pp. 1003–1016.

[9] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," in *IEEE Communications Magazine*, 2002, pp. 102–114.

[10] S. Aldosari and J. Moura, "Fusion in sensor networks with communication constraints," in *ACM/IEEE IPSN*, 2004, pp. 108–115.

[11] M. Ali, U. Saif, A. Dunkels, T. Voigt, K. Römer, and K. Langendoen, "Medium access control issues in sensor networks," in *ACM SIGCOMM CCR*, 2006, pp. 33–36.

[12] W. Baek, D. Wei, and C. Kuo, "Power-aware topology control for wireless ad-hoc networks," in *IEEE WCNC*, 2006, pp. 406–412.

[13] S. Basagni, R. Bruno, G. Mambrini, and C. Petrioli, "Comparative performance evaluation of scatternet formation protocols for networks of bluetooth devices," in *ACM/Kluwer Journal on Wireless Networks (WINET)*, vol. 10, 2004, pp. 197–213.

[14] M. Batalin and G. S. Sukhatme, "Coverage, exploration and deployment by a mobile robot and communication network," in *Telecommunication Systems Journal, Special Issue on Wireless Sensor Networks*, 2004.

[15] M. Burkhart, P. V. Rickenbach, R. Wattenhofer, and A. Zollinger, "Does topology control reduce interference?" in *ACM MobiHoc*, 2004, pp. 9–19.

[16] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat monitoring: Application driver for wireless communications technology," in *ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, 2001, pp. 20–41.

[17] C. Chang, K. Shih, H. Chang, and H. Liu, "Energy-balanced deployment and topology control for wsn," in *IEEE GlobeCom*, 2006, pp. 1–5.

[18] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *7th annual international conference on Mobile computing and networking*, 2001, pp. 85–96.

[19] X. Cheng, B. Narahari, R. Simha, M. Cheng, and D. Liu, "Strong minimum energy topology in wireless sensor networks: Np-completeness and heuristics," in *IEEE Trans. on mobile computing*, vol. 2, 2003, pp. 248–255.

[20] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher, "Real-time power-aware routing in sensor networks," in *IEEE IWQoS*, 2006, pp. 83–92.

[21] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, "On network correlated data gathering," in *IEEE INFOCOM*, 2004, pp. 2571–2582.

[22] F. Cuomo, T. Melodia, and I. F. Akyildiz, "Distributed self-healing and variable topology optimization algorithms for qos provisioning in scatternets," in *IEEE Journal on Selected Areas in Communications*, vol. 22, 2004, pp. 1220–1236.

[23] M. Dhanaraj, B. S. Manoj, and C. S. R. Murthy, "A new energy efficient protocol for minimizing multi-hop latency in wireless sensor networks," in *IEEE PerCom*, 2005, pp. 117–126.

[24] Q. Dong, "Maximizing system lifetime in wireless sensor networks," in *IPSN*, 2005, pp. 13–19.

[25] C. Drula, C. Amza, F. Rousseau, and A. Duda, "Adaptive energy conserving algorithms for neighbor discovery in opportunistic bluetooth networks," in *IEEE Transactions on Selected Areas in Communications*, vol. 25, 2007, pp. 96–107.

[26] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, "Geometric spanners for routing in mobile networks," in *ACM MobiHoc*, 2001, pp. 45–55.

[27] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher, "Range-free localization schemes in large scale sensor networks," in *ACM MobiCom*, 2003, pp. 81–95.

[28] K. Jamieson, H. Balakrishnan, and Y. Tay, "Sift: A mac protocol for event-driven wireless sensor networks," in *LCS-TR-894, MIT*, 2003.

[29] S. Jung, M. Gerla, C. Kalló, and M. Brunato, "Decentralized optimization of dynamic bluetooth scatternets," in *Mobiquitous*, 2005, pp. 305–313.

[30] S. Kardos and A. Vidács, "Performance of a new device discovery and link establishment protocol for bluetooth," in *IEEE Globecom*, 2005, pp. 3518–3522.

[31] A. Keshavarzian, H. Lee, and L. Venkatraman, "Wakeup scheduling in wireless sensor networks," in *ACM MobiHoc*, 2006, pp. 322–333.

[32] B. Krishnamachari, *Networking Wireless Sensors*, 2006.

[33] B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *International Workshop on Distributed Event-based Systems*, 2002.

[34] K. Langendoen and G. Halkes, *Embedded Systems Handbook*. CRC Press, 2005, ch. Energy-Efficient Medium Access Control.

[35] C. Law and K. Siu, "A bluetooth scatternet formation algorithm," in *IEEE Globecom*, 2001, pp. 2864–2869.

[36] J. Li and P. Mohapatra, "An analytical model for the energy hole problem in many-to-one sensor networks," in *IEEE VTC*, 2005, pp. 2721–2725.

[37] L. Li and J. Halpern, "Minimum energy mobile wireless networks revised," in *IEEE ICC*, 2001, pp. 278–283.

[38] N. Li, J. Hou, and L. Sha, "Design and analysis of an mstbased topology control algorithm," in *IEEE INFOCOM*, 2003, pp. 1702–1712.

[39] X. Li, I. Stojmenovic, and Y. Wang, "Partial delaunay triangulation and degree limited localized bluetooth scatternet formation," in *Parallel and Distributed Systems*, 2004, pp. 350–361.

[40] J. Lian, K.Naik, and G. B. Agnew, "Modeling and enhancing data capacity in wireless sensor networks," in *IEEE Monograph on Sensor Network Operations*, 2004.

[41] J. Lian, K. Naik, and G. B. Agnew, "Data capacity improvement of wireless sensor networks using non-uniform sensor distribution," in *Intern. Journal of Distr. Sensor Networks*, 2005.

[42] G. Lu, B. Krishnamachari, and C. Raghavendra, "An adaptive energy-efficient and low-latency mac for data gathering in wireless sensor networks," in *IEEE IPDPS*, 2004, pp. 224–231.

[43] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay efficient sleep scheduling in wireless sensor networks," in *IEEE Infocom*, 2005, pp. 2470–2481.

[44] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *IEEE WSNA*, 2002, pp. 88–97.

[45] M. Marsan, C. Chiasserini, and A. Nucci, "Forming optimal topologies for bluetooth-based wireless personal area networks," in *IEEE Transactions on Wireless Communications*, vol. 5, 2006, pp. 763–773.

[46] M. Medidi and J. Campbell, "Energy-efficient bounded-diameter tree scatternet for bluetooth pans," in *IEEE LCN*, 2005, pp. 268–275.

[47] M. Medidi, R. Slaaen, Y. Zhou, C. Mallery, and S. Medidi, "Scalable localization in wireless sensor networks," in *IEEE HiPC*, vol. LNCS 4297, 2006, pp. 522–533.

[48] K. Moaveni-Nejad and X. Li, "Low-interference topology control for wireless ad hoc networks," in *Ad Hoc & Sensor Wireless Networks: An International Journal*, 2005.

[49] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *SenSys*, 2004, pp. 250–262.

[50] S. Olariu and I. Stojmenovic, "Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting," in *IEEE INFOCOM*, 2006, pp. 1–12.

[51] S. Pack, J. Choi, T. Kwon, and Y. Choi, "Ta-mac: Task aware mac protocol for wireless sensor networks," in *IEEE VTC*, 2006, pp. 294–298.

[52] C. Pamuk and E. Karasan, "Sf-devil: an algorithm for energy-efficient bluetooth scatternet formation and maintenance," in *Computer Communications*, vol. 28, 2005, pp. 1276–1291.

[53] M. Perillo, Z. Cheng, and W. Heinzelman, "On the problem of unbalanced load distribution in wireless sensor networks," in *IEEE GlobeCom*, 2004, pp. 74–79.

[54] C. Petrioli, S. Basagni, and I. Chlamtac, "Configuring bluestars: multihop scatternet formation for bluetooth networks," in *IEEE Transactions on Computers 52(6), Spl. Issue on Wireless Internet*, 2003, pp. 779–790.

[55] C. Petrioli, S. Basagni, and I.Chlamtac, "Bluemesh: Degree-constrained multihop scatternet formation for bluetooth networks," in *Mobile Networks and Applications*, vol. 9, 2004, pp. 33–47.

[56] S. Ping, "Delay measurement time synchronization for wireless sensor networks," in *Intel Research*, vol. IRB-TR-03-013, 2003.

[57] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," in *IEEE ICRA*, 2004, pp. 165–171.

[58] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *ACM SenSys*, 2004, pp. 95–107.

[59] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," in *ACM SenSys*, 2003, pp. 181–192.

[60] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-mac: a hybrid mac for wireless sensor networks," in *ACM SenSys*, 2005, pp. 90–101.

[61] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," in *IEEE JSAC*, 1999, pp. 1333–1344.

[62] K. Römer, "Time synchronization and localization in sensor networks," Ph.D. dissertation, Swiss Federal Institute of Technology Zurich, 2005.

[63] R. Roy, M. Kumar, N. K. Sharma, and S. Sural, "Bottom-up construction of bluetooth topology under a traffic-aware scheduling scheme," in *IEEE Transactions on Mobile Computing*, vol. 6, 2007, pp. 72–86.

[64] S. Saginbekov and I. Korpeoglu, "An energy efficient scatternet formation algorithm for bluetooth-based sensor networks," in *2nd European Workshop on Wireless Sensor Networks*, 2005, pp. 207–216.

[65] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, "Distributed topology construction of bluetooth personal area networks," in *IEEE INFOCOM*, 2001, pp. 1577–1586.

[66] P. Santi, "Topology control in wireless ad hoc and sensor networks," in *ACM Computing Surveys,*, vol. 37, 2005, pp. 164–194.

[67] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing sensor networks in the energy-latency-density design space," in *IEEE Transactions on Mobile Computing*, 2002, pp. 70–80.

[68] W. Song, Y. Wang, X. Li, and O. Frieder, "Localized algorithms for energy efficient topology in wireless ad hoc networks," in *ACM MobiHoc*, 2004, pp. 98–108.

[69] S. Soro and W. Heinzelman, "Prolonging the lifetime of wireless sensor networks via unequal clustering," in *IEEE IPDPS*, 2005.

[70] S. Soro and W. B. Heinzelman, "Prolonging the lifetime of wireless sensor networks via unequal clustering," in *IEEE IPDPS*, 2005, pp. 8–15.

[71] I. Stojmenović, "Dominating set based bluetooth scatternet formation with localized maintenance," in *IEEE IPDPS*, 2002, pp. 148–155.

[72] S. Sunkavalli and B. Ramamurthy, "Mtsf: a fast mesh scatternet formation algorithm for bluetooth networks," in *IEEE Globecom*, 2004, pp. 3594–3598.

[73] G. Tan, A. Miu, J. Guttag, and H. Balakrishnan, "An efficient scatternet formation algorithm for dynamic environments," in *IASTED CCN*, 2001.

[74] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *ACM SenSys*, 2003, pp. 171–180.

[75] L. van Hoesel and P. Havinga, "A lightweight medium access protocol (lmac) for wireless sensor networks," in *1st Int. Workshop on Networked Sensing Systems (INSS 2004)*, 2004.

[76] L. van Hoesel, T. Nieberg, H. Kip, and P. Havinga, "Advantages of a tdma based, energyefficient, self-organizing mac protocol for wsns," in *IEEE VTC*, 2004, pp. 1598–1602.

[77] M. Vuran and I. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks," in *IEEE/ACM Transactions on Networking*, vol. 14, 2006, pp. 316–329.

[78] H. Wang, X. Zhang, and A. Khokhar, "An energy-efficient low-latency mac protocol for wireless sensor networks," in *IEEE GLOBECOM*, 2006, pp. 1–5.

[79] S. Wang, D. Wei, and S. Kuo, "Spt-based power-efficient topology control for wireless ad hoc networks," in *IEEE MILCOM*, 2004, pp. 1483–1490.

[80] Y. Wang and X. Li, "Distributed spanners with bounded degree for wireless ad hoc networks," in *IEEE IPDPS*, 2002, pp. 194–201.

[81] Z. Wang, R. J. Thomas, and Z. Haas, "Bluenet-a new scatternet formation scheme," in *35th HICSS*, 2002.

[82] X. Wu, G. Chen, and S. K. Das, "On the energy hole problem of nonuniform node distribution in wireless sensor networks," in *IEEE MASS*, 2006, pp. 180–187.

[83] Y. Xu, S. Bien, Y. Mori, J. Heidemann, and D. Estrin, "Topology control protocols to conserve energy in wireless ad hoc networks," UCLA, Tech. Rep., 2003.

[84] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *7th annual international conference on Mobile computing and networking*, 2001, pp. 70–84.

[85] X. Yang and N. Vaidya, "A wakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay," in *IEEE RTAS*, 2004, pp. 19–26.

[86] M. Ye, C. Li, G. Chen, and J. Wu, "Eecs: an energy efficient clustering scheme in wireless sensor networks," in *IEEE IPCCC*, 2005, pp. 535–540.

[87] W. Ye, J. Heidemann, and D.Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," in *IEEE/ACM Transactions on Networking*, 2004, pp. 493–506.

[88] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *IEEE Infocom*, 2002, pp. 1567–1576.

[89] Y.Wang and X. Li, "Efficient construction of bounded degree and planar spanner for wireless networks," in *ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, 2003, pp. 374–384.

[90] G. Záruba, S. Basagni, and I. Chlamtac, "Bluetrees - scatternet formation to enable bluetooth-based ad hoc networks," in *IEEE ICC*, 2001, pp. 273–277.

[91] H. Zhang, J. C. Hou, and L. Sha, "Bluetooth loop scatternet formation algorithm," in *IEEE ICC*, 2003, pp. 1174–1180.

[92] X. Zhang and G. F. Riley, "An on-demand bluetooth scatternet formation and routing protocol for wireless sensor networks," in *Software Engg., AI., Networking & Parallel/Distributed Computing*, 2005, pp. 411–418.

[93] Y. Zhang, S. Liu, W. Jia, and X. Cheng, "Bluepower - a new distributed multihop scatternet formation protocol for bluetooth networks," in *IEEE ICPP*, 2005, pp. 287–294.

[94] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *ACM Sensys*, 2003, pp. 1–13.

[95] M. Zorzi and R. Rao, "Geographic random forwarding (geraf) for ad hoc and sensor networks: energy and latency performance," in *IEEE Transactions on Mobile Computing*, vol. 2, 2003, pp. 349–365.