DUTY CYCLING FOR ENERGY EFFICIENCY IN WIRELESS SENSOR NETWORKS AND APPLICATIONS

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Jing Li, B.S., M.S.

Graduate Program in Department of Computer Science & Engineering

The Ohio State University

2012

Dissertation Committee:

Anish Arora, Advisor
Ten-Hwang Steve Lai
Ness B Shroff
Abstract

Wireless sensor networks (WSNs) offer a powerful combination of distributed sensing, computing and communication, which enable a broad spectrum of applications and, at the same time, lead numerous challenges due to their distinctiveness, primarily the non-negligible power consumption from especially radio activities and stringent energy constraints to which sensor nodes are typically subjected. The distinguishing traits of sensor networks have a direct impact on their protocol design at each layer, especially at the Medium Access Control (MAC) layer since it manages transmission scheduling as well as duty cycling for energy conservation. To maximize energy efficiency of WSNs, my thesis studies duty cycling in time and frequency domains for both MAC schedulers and applications. The first part of the thesis focuses on energy efficiency at the MAC layer, including modeling, evaluating and designing MAC schedulers with duty cycling; the second part of the dissertation investigates energy efficiency in applications, which introduces two duty-cycled sensor applications that are deployed in a large building.

In the first part of this dissertation, I begin by studying the impact of perfect duty cycling, in addition to perfect transmission scheduling, on the capacity of random wireless networks with single and multiple channels. The analysis of the duty-cycled throughput reveals nontrivial scaling gains resulting from the ability to avoid interference by spreading interferers to mutually exclusive times, which corroborates
the importance of efficient co-scheduling of both transmissions and duty cycling for energy efficiency. Since duty cycling and transmission scheduling are controlled by MAC schedulers, I analytically quantify the gap between the duty-cycled throughput with an optimal scheduler and with existing MAC schedulers.

In order to characterize energy efficiency achieved by existing MACs, I classify CSMA-based MAC protocols in terms of critical MAC-design factors into four classes and introduce an analytical framework for performance modeling of each class as a function of key protocol parameters. I instantiate the framework to evaluate various performance metrics of MACs across the configuration space. A surprising finding is that one MAC class consistently achieves the best or close-to-the-best performance across much of the configuration space. Moreover, via the analytical model, I discover a distributed way of adapting duty cycle at the MAC layer to changing traffics for optimality of performance. In terms of energy efficiency in the frequency domain, I propose Chameleon, which is a light-weight MAC protocol that maximizes energy efficiency over the spectrum by scheduling traffics across multiple frequencies with duty cycling.

In the second part of this dissertation, I present two long-lived sensor networks deployed in a large building. Towards duty cycling Heating, Ventilation, and Air Conditioning systems of large buildings such that comfort and efficiency can be maintained simultaneously, I described ThermoNet, which is a system for temperature monitoring in large buildings. Access to fine grain information reveals temporal and spatial dynamics that help quantify the level of (non-)compliance with the building’s thermal comfort standards and identify ill-conditioned rooms that need maintenance. For another application, to increase the battery life of the elevator network from a couple of
days to several years, I introduce a self-stabilizing token-ring protocol that maintains
duty-cycle coordination across the partitions of a static network of nodes.
This is dedicated to my husband Dapeng Wang, my mother Zhongsha Tang and father Yuchang Li
Acknowledgments

It would be impossible to complete my Ph.D. study without guidance and help from many great people in the past six years.

I would like to thank my advisor Prof. Anish Arora for his guidance and support throughout the production of this research and dissertation. His insightful advice and positive attitude have impacted me deeply in my research and life. He always encourages me to think new ideas and helps me transform these ideas into real work. I greatly appreciate his patience in guiding me through the road to high quality research, from defining problems to explaining results.

I would also like to thank Prof. Ten-Hwang Lai, Prof. Ness B. Shroff, and Prof. Dong Xuan for their helpful suggests and for serving on my candidacy and dissertation committee. Prof. Ten-Hwang Lai has taught me the basics of algorithm theory which help me a lot in my research. Prof. Ness B. Shroff has provided me insightful suggestion during my candidacy exam. Prof. Dong Xuan has taught me a few major courses and provided me many helps in my study and research.

It is my fortunate to know many professors in the CSE department of OSU. Prof. David Lee recommended me for university fellowship for my first year study at OSU and gave me the opportunity to join his research group on protocol testing and security. Although I chose to work on wireless sensor networks after the first year, I have owed him a lot for his advise and support. Prof. Prasun Sinha taught
me several courses in my area of study and advised me patiently on my first course project. Prof. Srinivasan Parthasarathy is the advisor of one of my minors. It is him who guided me to the area of data mining. The knowledge I have learnt in his classes has been applied extensively in my research.

In the past six years, I have been involved in several NSF funded research programs. All of them are full of joy and excitement. I have benefited from many great scientists and researchers including Dr. William Leal, Dr. Rajiv Ramnath, Dr. Sandeep Kulkarni, and Dr. Emre Ertin.

I would also like to thank my fellow lab members, past and present, for their assistance, support, and friendship over the years: Dr. Lifeng Sang, Dr. Hui Cao, Dr. Hongwei Zhang, Dr. Vinayak Naik, Dr. Vinodkrishnan Kulathumani, Dr. Mukundan Sridharan, Dr. Taewoo Kwon, Dr. Sandip Bapat, Wenjie Zeng, Jin He, Ingy Youssef, Debraj De, Michael McGrath, Dong Zhang, Mitesh Kanjariya, Yue Qiao, and Balaji Palaniswami. Dr. Lifeng Sang gave me many valuable advise on my study and shared with me his experience in doing research. Dr. Mukundan Sridharan and Jin He contributed a lot to the development and maintenance of PeopleNet and ThermoNet especially when I was out of town for internship. Wenjie Zeng and I have been working on several topics together in these years. Without him, it would be impossible to complete the experimental validation of MAC protocols in my thesis at this time.

I would like to thank my mother, Zhongsha Tang, and my father, Yuchang Li for their love and support during these years. They have always encouraged me towards excellence. Finally, I would like to thank my loving husband, Dapeng Wang, for
sticking with me through all the good times and bad, and supporting me to persevere with the study.
Vita

September 2, 1983 ........................... Born - Anhui, China

2003 ................................. B.S. Computer Science,
Department of Computer Science,
Hefei University of Technology.

2006 ................................. M.S. Computer Science,
Department of Computer Science,
University of Science & Technology of
China.

July-September 2010 ..................... Research Intern,
Corporate Development Technology
Group,
Cisco Inc., San Jose, USA

2006-present .............................. Graduate Teaching & Research Asso-
ciate,
Department of Computer Science &
Engineering,
The Ohio State University.

Publications

Research Publications


Fields of Study

Major Field: Computer Science and Engineering
Studies in:

Computer Networking  Prof. Anish Arora  
Prof. Ten-Hwang Lai  
Prof. Ness B. Shroff  
Prof. Dong Xuan  
Prof. Prasun Sinha  
Prof. David Lee  

Data Mining  Prof. Srinivasan Parthasarathy  
Prof. Hakan Ferhatosmanoglu  

Theory and Algorithms  Prof. Simge Kucukyavuz  
Prof. Emre Koksal  

xi
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>vi</td>
</tr>
<tr>
<td>Vita</td>
<td>ix</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xvi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xvii</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Applications of Sensor Networks</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Characteristics of Sensor Networks</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Energy Efficiency</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Contributions</td>
<td>6</td>
</tr>
<tr>
<td>1.5 Organization of this Thesis</td>
<td>11</td>
</tr>
<tr>
<td>2. Achievable Throughput in Duty-Cycled Wireless Networks</td>
<td>13</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>14</td>
</tr>
<tr>
<td>2.1.1 Summary of the Results</td>
<td>15</td>
</tr>
<tr>
<td>2.1.2 Organization of this Chapter</td>
<td>16</td>
</tr>
<tr>
<td>2.2 Related Work</td>
<td>16</td>
</tr>
<tr>
<td>2.3 Duty-Cycled Capacity of Single-Channel Wireless Networks</td>
<td>17</td>
</tr>
<tr>
<td>2.3.1 System Model</td>
<td>17</td>
</tr>
<tr>
<td>2.3.2 Capacity of Multi-Hop Traffic</td>
<td>18</td>
</tr>
<tr>
<td>2.3.3 Capacity of Single-Hop Traffic</td>
<td>23</td>
</tr>
<tr>
<td>2.4 Duty-Cycled Capacity of Multi-Channel Wireless Networks</td>
<td>25</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>4.3</td>
<td>Energy Efficiency Analysis</td>
</tr>
<tr>
<td>4.3.1</td>
<td>System Model</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Energy Efficiency Optimization</td>
</tr>
<tr>
<td>4.4</td>
<td>Energy Efficient Multi-Channel Protocol Design</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Channel to Load Assignment</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Receiver-Centric Wakeup and Channel Scheduler</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Implementation</td>
</tr>
<tr>
<td>4.5</td>
<td>Protocol Evaluation</td>
</tr>
<tr>
<td>4.6</td>
<td>Conclusions</td>
</tr>
<tr>
<td>5</td>
<td>ThermoNet: Fine-Grain Assessment of Building Comfort and Efficiency</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Summary of the Results</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Organization of this Chapter</td>
</tr>
<tr>
<td>5.2</td>
<td>Related Work</td>
</tr>
<tr>
<td>5.3</td>
<td>Sensor Network System Design</td>
</tr>
<tr>
<td>5.3.1</td>
<td>System Architecture</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Dynamic Power Adaptation</td>
</tr>
<tr>
<td>5.4</td>
<td>Thermal Comfort Assessment</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Building Comfort Level</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Seasonal Mode Change Anomaly</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Long-Term Ill-Conditioning</td>
</tr>
<tr>
<td>5.4.4</td>
<td>Daily Anomaly Detection</td>
</tr>
<tr>
<td>5.5</td>
<td>HVAC System Efficiency Assessment</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Reducing On-going Over-Conditioning</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Longer Temperature Setbacks</td>
</tr>
<tr>
<td>5.6</td>
<td>Data-Driven Thermal Model</td>
</tr>
<tr>
<td>5.6.1</td>
<td>Cluster Identification</td>
</tr>
<tr>
<td>5.6.2</td>
<td>Cluster-Based Modeling</td>
</tr>
<tr>
<td>5.6.3</td>
<td>Correlation of Responses of Cluster Heads</td>
</tr>
<tr>
<td>5.7</td>
<td>Conclusions</td>
</tr>
<tr>
<td>6</td>
<td>Duty Cycle Stabilization in Semi-Mobile Wireless Networks</td>
</tr>
<tr>
<td>6.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Duty Cycling for Elevator Sensing</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Summary of the Results</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Organization of this Chapter</td>
</tr>
<tr>
<td>6.2</td>
<td>Related Work</td>
</tr>
<tr>
<td>6.3</td>
<td>System Model and Problem Statement</td>
</tr>
<tr>
<td>6.4</td>
<td>Duty-Cycle Coordination in a Multi-Token Ring</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 MAC protocols of different classes</td>
<td>30</td>
</tr>
<tr>
<td>2.2 Model parameters</td>
<td>34</td>
</tr>
<tr>
<td>2.3 MAC dependent parameters for four MAC schedulers</td>
<td>36</td>
</tr>
<tr>
<td>3.1 Known MAC protocol performance comparisons</td>
<td>48</td>
</tr>
<tr>
<td>3.2 MAC-specific parameters in the framework</td>
<td>57</td>
</tr>
<tr>
<td>3.3 Notation and typical values of constants</td>
<td>59</td>
</tr>
<tr>
<td>5.1 Thermal model of selected cluster heads and their root mean square errors of 4-fold cross validation.</td>
<td>142</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Analytical comparison of per node throughput at diverse network configurations in a clique.</td>
<td>39</td>
</tr>
<tr>
<td>2.2</td>
<td>Experimental comparison of per node throughput at diverse network configurations in a clique.</td>
<td>43</td>
</tr>
<tr>
<td>3.1</td>
<td>Comparison of MAC (a) delivery ratio and (b) duty cycle at diverse network configurations.</td>
<td>70</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparison of MAC (a) energy efficiency and (b) energy efficiency at latency $T^\ast$.</td>
<td>71</td>
</tr>
<tr>
<td>3.3</td>
<td>Tradeoff between queue size and energy efficiency for O-MAC.</td>
<td>73</td>
</tr>
<tr>
<td>3.4</td>
<td>Performance of SCP-MAC under various queue sizes (a) delivery ratio and (b) energy efficiency.</td>
<td>74</td>
</tr>
<tr>
<td>3.5</td>
<td>Performance of O-MAC under various queue sizes (a) delivery ratio and (b) energy efficiency.</td>
<td>75</td>
</tr>
<tr>
<td>3.6</td>
<td>Analytical performance of O-MAC, BoX-MAC, and RI-MAC at diverse network configurations (a) delivery ratio and (b) energy efficiency.</td>
<td>77</td>
</tr>
<tr>
<td>3.7</td>
<td>Experimental performance of O-MAC, BoX-MAC, and RI-MAC at diverse network configurations (a) delivery ratio and (b) energy efficiency.</td>
<td>78</td>
</tr>
<tr>
<td>3.8</td>
<td>Percentage change in energy efficiency when senders become hidden terminals.</td>
<td>81</td>
</tr>
<tr>
<td>3.9</td>
<td>Optimal frame lengths for O-MAC versus flow rates under different network densities.</td>
<td>82</td>
</tr>
</tbody>
</table>
3.10 Performance reduction with standard deviation due to local frame length adaptation scheme for O-MAC (a) delivery ratio reduction and (b) energy efficiency reduction. 

3.11 Diagram of spanning tree network topology. 

4.1 (a) $\hat{E}(k)$ vs $\bar{p}(k)$ and (b) Mean $w(k)$ with std vs Metric $\sum p_j(k)$ 

4.2 Mean error of measured $I(k)$ versus the duty cycle of measurement. 

4.3 Composition of the Chameleon protocol. 

4.4 The number of internal network flows increases in an experimental clique network, (a) average packet delivery ratio, (b) average duty cycle, (c) energy efficiency. External interference load changes in an experimental clique network (d) average packet delivery ratio, (e) average duty cycle, (f) energy efficiency. 

5.1 Picture of (a) Dreese Laboratory at OSU and (b) ThermoNet architecture illustrated for the 2nd floor of Dreese Laboratory. 

5.2 Picture of (a) sensor node, exposed (left) and in its enclosure as deployed in rooms of Dreese (middle and right). (b) state diagram of power adaptation algorithm in ThermoNet. 

5.3 Picture of (a) average supply voltage with std at Tier 3 nodes between June 2010 and May 2011. (b) Probability Mass Function of transmission power levels used by Tier 3 nodes to achieve 98% PRR over the year. 

5.4 Picture of (a) daily ratios of hot, comfortable, and cold rooms in Dreese. (b) Long-term indoor and outdoor weather comparison. 

5.5 Picture of (a) Cumulative Distribution Function for percentage of days over a year that a room has been ill-conditioned. (b) Classification of Dreese rooms — Red: persistently hot; Blue: persistently cold; Pink: intermittently hot; Cyan: intermittently cold; Green: borderline or well-conditioned rooms.
5.6 Anomalous rooms on Dec. 22, 2010, as identified by DBSCAN clustering algorithm (epsilon=0.15, minPoints=6). Rooms that reside outside the dotted oval are anomalies. ........................................ 136

5.7 Picture of (a) indoor hourly temperatures of a day for four seasons. (b) Estimated percentages of occupied rooms for each hour of a day during one quarter and the break preceding it. ...................... 139

5.8 Visualization of clustering result based on every 15-min temperature. 141

5.9 Picture of (a) experimental (solid) and simulated (dashed) average building temperature (\(\hat{T}\)), (b) experimental (solid) and simulated (dashed) average temperature deviation of all rooms from 72\(^\circ\)F (\(T_{dev}\)). .... 143

5.10 CDF for the standard deviation of temperature changes among cluster heads under the same control inputs. .................. 144

6.1 Picture of (a) a Dreese elevator and (b) TelosB mote deployed in the elevator. .......................... 147

6.2 A snapshot of PeopleNet elevator localization webpage. ............. 148

6.3 Illustration of continuous synchronization ......................... 156
Chapter 1: INTRODUCTION

Wireless Sensor Network (WSN) comprises tens to thousands of autonomic and self-organizing sensor devices that co-operate with each other for environmental monitoring, data processing, and wireless networking. In addition to traditional endeavor to balance throughput, latency, and fairness concerns, protocols of WSN place an emphasis on energy efficiency due to non-negligible power consumption from especially radio activities and limited energy resources carried by individual node. This chapter provides background knowledge on wireless sensor network applications and characteristics, summarizes the main contributions of this dissertation, and introduces the organization of the thesis.

1.1 Applications of Sensor Networks

Wireless sensor networks consist of a large number of nodes, which can sense their environment over time, communicate with each other over a wireless channel, and process information either locally or at a gateway. The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance, where sensor nodes equipped with processing and communication capabilities can collaboratively detect, classify, and track targets of interest over a large area. The project ExScal (for Extreme Scale) [14] fielded a 1000+ node wireless sensor
network and a 200+ node peer-to-peer ad-hoc network of 802.11 devices in a 1.3km by 300m remote area in Florida, USA during December 2004. The ExScal concept of operation is to deploy a dense wireless sensor network “tripwire” that detects, tracks, and classifies multiple intruders of different types (such as people and vehicles) in a long perimeter region. Application of this concept is envisioned for protection of pipelines that are vulnerable to sabotage, borders between nations that are prone to illegal crossing, and areas abutting critical plants/thoroughfares that are vulnerable to terrorist threat.

Sensor networks are also applied in many civilian application areas, including environment and habitat monitoring, medical research, healthcare, and etc. Joint efforts between Harvard University, the University of New Hampshire, and the University of North Carolina led to the deployment of a wireless sensor network to monitor eruptions at Volcano Tungurahua, an active volcano in central Ecuador. A network of Berkeley motes monitored infrasonic signals during eruptions, and data were transmitted over a 9 km wireless link to a base station at the volcano observatory [64]. In medical research and healthcare area, wearable sensors can be used to capture vital signs from patients in real-time and relay the data to handheld computers carried by medical personnel. With these ideas in mind, the Ohio State University cooperated with other institutions and developed AutoSense, a unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field. It was applied in a 20+ subject real-life scientific study on stress in both the lab and field with high accuracy [30].

Another promising WSN application in civil engineering is the idea of Smart Buildings: wireless sensor and actuator networks integrated within buildings could allow
distributed monitoring and control, improving living conditions and reducing the energy consumption by controlling set point and air flow of the HVAC system. The Berkeley LoCal project aims to produce a network architecture for localized electrical energy reduction, generation, and sharing by examining how pervasive information can fundamentally change the nature of these processes [15]. They implement sensing data-driven control strategies on HVAC testbed for energy reduction. A key component of this project is the use of sMAP [24] to exchange physical data about the systems involved, which allows producers of physical information to directly publish their data in a format for consumption by a diverse set of clients. The last two chapters of this dissertation focus on two smart building applications that we deployed inside our departmental building, Dreese Laboratories.

1.2 Characteristics of Sensor Networks

Compared to the traditional computer networks, wireless sensor networks have several unique characteristics as listed in [60].

- Application specific: A WSN is deployed to perform a specific task, e.g. environmental monitoring, target tracking or alerting. As a result, the node platforms and communication protocols are designed to optimal performance on a certain application-dependent scenario. The application specific behavior enables data aggregation, in-network processing and decision making.

- Low cost: To allow cost effective deployment of a large quantity of nodes, the cost of an individual sensor node should be minimized. Also, as recovering sensors after deployment in some application scenarios may not be feasible, sensors should be cheap enough to be considered disposable.
• Resource constraint: A typical WSN node combines low cost with small physical size and is battery powered. Thus, their sensing, processing, storing, communication capabilities, and energy resources are very limited.

• Dynamic nature: Wireless communications are inherently unreliable due to environmental interferences. The unreliability is especially evident in WSNs because of harsh operating conditions, e.g., due to environmental changes in outdoors, node mobility, and nodes dying due to depleted energy sources. As a result, the unreliability causes network dynamics due to link breaks even when nodes are stationary.

• Network lifetime: WSNs are typically deployed to observe certain physical phenomenon that range in duration from fractions of a second to a few months or even several years. As replacing batteries is not feasible due to large network size and deployment to possibly hazardous environment, sensor nodes must optimize their energy usage for network lifetime.

1.3 Energy Efficiency

Sensor networks provide endless opportunities, but at the same time pose formidable challenges, such as the fact that energy is a scarce and usually non-renewable resource. In practice, it will be necessary in many applications to provide guarantees that a network of unattended wireless sensors can remain operational without any replacements for several years [43]. Hardware improvements in battery design and energy harvesting techniques will offer only partial solutions. This motivates protocol designs in wireless sensor networks with explicit energy efficiency as the primary goal. Since radio communication is the dominant power consumption in all the components, most
energy-efficient protocols depend on duty cycling, which periodically turn on and off the radios to conserve energy. Naturally, this goal must be tradeoff against a number of other concerns. Obviously, synchronization of such schedules becomes challenging since nodes may miss control messages from each other while the radio is switched off. The problem turns to be even harder when multiple frequencies are explored simultaneously with duty cycling since periodic radio-off leads to nontrivial difficulty in both time and channel rendezvous. Aside from synchronization issue, an arbitrarily long sleep periods can also reduce the responsiveness and effectiveness of the sensors. In applications that require rapid detection of events in the environment, the latency induced by sleep schedules must be kept within strict bounds. In this dissertation, we investigate these tradeoffs of duty cycling in MAC protocols across a wide range of network settings.

One primary goal of the thesis is to characterize energy efficiency of MAC protocols, which is defined as the ratio of the useful communication energy expended to the total energy expended [21]. Basically, this metric refers to the ratio of the number of time slots with successful receptions to the number of slots in which radios are active, albeit they are transmitting, idle, or active. For instance, Eq. (1.1) defines energy efficiency for a unicast scenario [21], where time is divided into contiguous slots and notation $T$ in the formula is the total number of slots considered.

$$E_E = \frac{\sum_{l=1}^{T} \sum_{j=1}^{N} Z_j^l}{\sum_{l=1}^{T} \sum_{j=1}^{N} (S_j^l + R_j^l)}$$

(1.1)
where

\[ S^l_j = \begin{cases} 1, & \text{when node } j \text{ transmits in slot } l \\ 0, & \text{when node } j \text{ sleeps in slot } l \end{cases} \]

\[ R^l_j = \begin{cases} 1, & \text{when node } j \text{ listens in slot } l \\ 0, & \text{when node } j \text{ sleeps in slot } l \end{cases} \]

\[ Z^l_j = \begin{cases} 2, & \text{node } j \text{ succeeds receiving its packets in slot } l \\ 0, & \text{otherwise} \end{cases} \]

1.4 Contributions

The main contributions of this dissertation are summarized in six aspects.

Achievable Throughput When Duty Cycling

Although a decade of productive research has been conducted in low power wireless networks, the duty-cycled throughput capacity, defined as the maximum rate in terms of number of nodes, \( n \), and their duty cycles, \( \psi \), is not easily computed, and methods for its estimation have not received much attention. My analysis shows that it is feasible to achieve nontrivial scaling gains in duty-cycled networks, i.e., avoid interference and conserve energy at the same time, by properly co-scheduling both transmissions and duty cycling. Since duty cycling is normally a function of the MAC layer, I analytically model and evaluate how well canonical classes of duty-cycled MAC protocols achieve throughput capacity with respect to what is achievable.

I show that in a multi-hop random network setting the duty-cycling of radios results in a per-node throughput capacity of \( \Theta(W \psi \sqrt{\log n} / \sqrt{n}) \), as opposed to \( \Theta(W \psi / \sqrt{n \log n}) \), where \( W \) denotes the maximum transmission rate of a node. Likewise, in a single-hop clique network the duty-cycling of radios results in a per-node throughput capacity of
Θ(Wψ), as opposed to Θ(Wψ/n). Similar results hold for multi-hop networks with multiple frequencies. These capacity gains of Θ(log n) and Θ(n) respectively result from spreading interference over time, indicating the importance of efficient co-scheduling of transmissions and sleep-wakeup, which is normally a function of the MAC layer. By analytically comparing duty-cycled capacities of four classes of MAC schedulers with that of the optimal, I find that out of the four classes receiver-centric synchronous scheme yields the smallest gap between existing MACs and the optimal scheduler.

**Energy Efficiency Achieved by MAC Protocols**

Conventional wisdom says that a MAC protocol that performs well for one application can perform poorly for another. Perhaps as a result of this view, the design and evaluation of MAC protocols has typically focused on relatively narrow portions of the configuration space. The configuration space includes not only the network configuration such as network density and traffic rate, but also the protocol configuration, a set of protocol parameters that determine how a protocol operates. *To characterize energy efficiency and the associated tradeoff of state-of-the-art MAC protocols across the configuration space, I present an analytical framework that can be instantiated for performance modeling of each MAC class as a function of key protocol parameters; extensive experimentation corroborates our analysis. A surprising finding of our comparative study is that the receiver-centric synchronous MAC class consistently achieves the best or close to the best performance for various metrics across much of the configuration space.*

The analysis and experiments show that across a rich set of network configurations, a receiver-centric synchronous protocol (O-MAC) outperforms other MAC classes in many aspects, including energy efficiency, throughput, latency and queue size.
This is because staggering receiver wakeup times and synchronizing nodes locally not only avoid generating unnecessary communication overhead (such as preambles and beacons), but also make communications more uniform and independent, which is preferable in general networks.

**Dynamic Control of Duty Cycling**

I show in the comparative study that adapting duty cycle to network dynamics is a critical factor for optimality of the MAC performance. The MAC modeling framework offers a systematic way of exploring optimal duty cycles under various network configurations. *Based on the instantiated framework, I identify the relationship between the frame length of a node and the incoming traffic rate towards the node. This yields a simple but effective way of adapting duty cycle locally on sensor node with limited computation capability. The proposed duty cycle control scheme is shown to achieve close to the optimal performance over a broad range of the configuration space.*

I observe that the optimal frame length of O-MAC is inversely proportional to its flow rate, as well as the number of flows, in a non-linear fashion. Notably, at different flow counts, the respective trends of the optimal frame length vary only slightly, suggesting that the optimal frame length can be modeled as a function of flow rate in a manner that is insensitive to the number of flows in the neighborhood. The frame length adaptation function for O-MAC is $T_f = C_1 / r_{pkt}$, where $C_1$ is a constant determined by the upper bound of frame lengths and maximal queue size; $r_{pkt}$ represents the total incoming traffic rate destined to a receiver. Alternatively speaking, each receiver may independently choose its frame length rather than rely
on the aggregate traffic in the neighborhood to achieve close-to-optimal performance. The effectiveness of this adaptation scheme is verified by the numerical results.

**Energy Efficiency in Multi-Channel MAC Protocol**

Towards improving energy efficiency in low-power wireless networks when multiple channels are available, I show that in this setting maximization of MAC energy efficiency reduces to maximizing the aggregate channel utilization and minimizing the aggregate duty cycle. *I describe a multi-channel MAC protocol, Chameleon, which practically realizes these schedulers in a distributed fashion with local information only. By assigning channels based on lightweight estimation of channel utilization and adapting the duty cycle of node reception in each channel, substantial energy efficiency gains are achieved by Chameleon over state-of-the-art multi-channel protocols.*

Chameleon optimizes MAC energy efficiency via two components: one for precisely quantifying utilization on each channel and the other for minimizing the duty cycle for a given node traffic. The first component passively computes at each receiver a light-weight metric $w$ that estimates each channel utilization by the sum of interference level and incoming traffic to the node. The incoming traffic is then split over different channels to maximize the aggregate channel utilization. The second component implements a receiver-centric synchronous scheduler for duty cycling, which has been shown to obtain the highest energy efficiency per channel. Experiments show that Chameleon intrinsically accommodates external interference and achieves an average of 24% to 66% energy efficiency gains over other multi-channel protocols.
Assessment of Building Comfort and Efficiency

Given that a large portion of energy is used for Heating, Ventilation, and Air-Conditioning (HVAC) system in buildings, many efforts have been made to improve HVAC performance by instrumenting wireless sensor networks in civilian areas, out of which fine-gained assessment of HVAC system performance in buildings is a prerequisite for optimizing building energy efficiency. To address this problem, I evaluate the thermal comfort and efficiency of a relatively modern HVAC system in a large building based on high-fidelity environmental data collected via a WSN over 12 months. I identify ill-conditioned rooms that need maintenance, and find over-conditioning at multiple time scales which offers opportunities for reduced operating cost.

I present ThermoNet, which is a low-cost, large-scale WSN that enables fine grain evaluation of HVAC performance in legacy buildings while meeting a multi year-long lifetime requirement via duty cycling and adaptive power control. The evaluation shows that the comfortable area with respect to the control objective averages at only 47% of the building. A substantial opportunity for energy savings exists based on dynamics at different time scales. For instance, many rooms are overcooled (overheated, respectively) in summer (in winter, respectively). Daily patterns of building temperature and (illumination-based) occupancy also show that the AHUs can be shut for several more hours at night while still meeting the control objective. For refining feedback to the HVAC controller, I construct a data-driven thermal model which provides the aggregate temperature of the building using a small set of room temperatures.
Duty Cycle Stabilization in Semi-Mobile WSNs

Coordinating the duty cycles of nodes in low-power wireless networks raises challenging stabilization issues. For instance, an abstract duty-cycling problem that stems from the indoor application of wireless elevator localization involves how to stabilize wakeup coordination across a network of partitioned clusters of static nodes via mobile token nodes that move between the clusters. *I show a self-stabilizing protocol that addresses the maintenance of duty-cycle coordination across isolated static nodes and tolerates faults such as node failure and state corruption in the network. The result is an increase in the battery life of the elevator network from a couple of days to several years.*

For the abstract “unidirectional multi-token ring” model, where the static nodes cannot communicate with other static nodes but only via the mobile tokens, I design and prove a stabilizing protocol by which all nodes converge to being simultaneously up according to a pre-selected global duty cycle. The essential idea of the protocol is to let a leader token dictate the frame schedule to all nodes and tokens in the system. The protocol consists of four main components: leader election, continuous frame synchronization, false leader detection, and global reset. Three refinements of the basic protocol are also presented.

1.5 Organization of this Thesis

The rest of this thesis is organized as follows.

Chapter 2 presents the analytical results on achievable throughput capacity in duty-cycled random wireless networks with single channel and multiple channels.
Duty-cycled capacities of state-of-the-art MAC schedulers are compared with the optimal scheduler in both theory and experiments.

Chapter 3 introduces a unified approach to analytically model energy efficiency and the associated tradeoff of each MAC class as a function of key protocol parameters. Various performance metrics are evaluated among MAC classes across the configuration space of these protocols; extensive experimentation corroborates the analysis.

Chapter 4 proposes a multi-channel MAC protocol, Chameleon, which maximizes energy efficiency by assigning channels based on lightweight estimation of channel utilization and adapting the duty cycle of node reception relative to the incoming traffic.

Chapter 5 navigates from duty cycling MAC protocols to investigating the duty cycle of HVAC system in a large building. This chapter shows assessment of building comfort and efficiency based on building-wide high-fidelity environmental data collected via a wireless sensor network over 12 months.

Chapter 6 abstracts a token ring problem that stems from the application of wireless elevator localization and provides a fault-tolerate algorithm that maintains duty cycle coordination across the partitions of a static network of nodes via information carried by mobile “token” nodes.

Chapter 7 summarizes the results of the thesis, and gives pointers to future research that can be based on this work.
Chapter 2: ACHIEVABLE THROUGHPUT IN DUTY-CYCLED WIRELESS NETWORKS

Capacity studies for wireless networks typically focus on scheduling of transmissions but rarely on scheduling of radio sleep-wakeup. We show that in a multi-hop network setting the duty-cycling of radios results in a per-node throughput capacity of $\Theta\left(\frac{W\psi \sqrt{\log n}}{\sqrt{n}}\right)$, as opposed to $\Theta\left(\frac{W\psi}{\sqrt{n \log n}}\right)$, where $\psi$ is the fraction of time each node radio is active. Likewise, in a single-hop clique network the duty-cycling of radios results in a per-node throughput capacity of $\Theta(W\psi)$, as opposed to $\Theta\left(\frac{W\psi}{n}\right)$. These capacity gains of $\Theta(\log n)$ and $\Theta(n)$ respectively result from spreading interference over time. They emphasize the importance of efficient co-scheduling of transmissions and sleep-wakeup, which is normally a function of the MAC layer. We also examine how well canonical classes of duty-cycled MAC protocols achieve throughput capacity. In particular, we abstract four schedulers, each of which represents several well-used MAC schedulers, namely sender-centric synchronous (e.g. S-MAC, T-MAC, DW-MAC, SCP-MAC), receiver-centric synchronous (e.g. WiseMAC, O-MAC), sender-centric asynchronous (e.g. B-MAC, X-MAC, BoX-MACs), and receiver-centric asynchronous (e.g. RI-MAC). We compare the achievable throughput of these schedulers analytically; extensive experiments are then performed to corroborate our MAC analysis.
The findings strongly suggest that out of the four classes receiver-centric synchronous scheme yields the smallest gap between existing MACs and the optimal scheduler.

2.1 Introduction

The capacity gains of using relatively short wireless links instead of long wireless links, resulting from the greater spatial reuse of the spectrum, have been well studied. The seminal work on throughput capacity by Gupta and Kumar established the per-node throughput in the protocol model for a random network where sources and destinations are randomly chosen [34] to be $\Theta(\frac{W}{\sqrt{n \log n}})$ bps per node. Here, $W$ and $n$ denote the maximum transmission rate of a node and the number of nodes in a network, respectively. Most capacity results assume perfect scheduling of transmissions, yet few take into account any sort of scheduling of the duty cycle of radios. By duty cycling, we mean the regular switching on and off of radios in order to conserve energy, which is crucial to the longevity of low-power wireless networks, especially wireless sensor networks. When the duty cycle of each node in terms of fractional on time is $\psi$, $0 \leq \psi \leq 1$, one might naively expect the throughput capacity of duty-cycled networks to be $\Theta(\frac{W\psi}{\sqrt{n \log n}})$.

In this chapter, we study the impact of perfect duty cycling, in addition to perfect transmission scheduling, on the capacity of multi-hop as well as single-hop networks. In practice, every scheduler incurs some measure of control overhead. Since duty cycle and transmission scheduling are inter-related and controlled by the MAC layer, we are motivated to quantify the gap between the duty-cycled throughput with an optimal scheduler and with existing MAC schedulers. We then examine how well
the capacity gains of using duty-cycling are realized by existing link-level (i.e., MAC) schedulers.

2.1.1 Summary of the Results

- Our analysis of the duty-cycled throughput capacity shows that for single-channel random wireless networks, per node throughput in a \( n \)-node general network operating at a duty cycle of \( \psi \) increases not as \( \Theta\left(\frac{W\psi}{\sqrt{n \log n}}\right) \) but as \( \Theta(W \sqrt{\frac{\psi}{n}}) \) till the canonical capacity limit is reached, which achieves a \( \Theta(\log n) \) throughput gain. For a network with \( c \) channels, the duty-cycled throughput becomes \( \Theta(W \sqrt{\frac{\psi c}{n}}) \).

- And when the network is limited to a single-hop clique, the per-node throughput becomes \( \Theta(W\psi) \), i.e., with perfect scheduling of transmissions and duty cycling, each node can pretend as though it is operating in isolation. These scaling gains result from the ability to avoid interference by spreading interferers to mutually exclusive times. This emphasizes the importance of efficient co-scheduling of both transmissions and duty cycling in order to properly exploit the resulting throughput capacity.

- We abstract and introduce a modeling framework for four schedulers, each of which represents a unique class of MAC protocols in terms of time synchronization and coordination centricity of communications. Since scheduling in MAC is fundamentally about single-hop communications, we leverage our result regarding optimal duty-cycled throughput for single-hop networks. This allows us to compare the single-hop throughput of different schedulers with the optimal.
• Finally, the soundness of our MAC models are corroborated by extensive experiments on the Kansei testbed [10].

2.1.2 Organization of this Chapter

The rest of this chapter is structured as follows. Section 2.2 discusses related work. Section 2.3 presents the main analytical results on duty-cycled throughput for random wireless networks. Section 2.4 extends the results for multi-channel and single-interface model. Section 2.5 proposes a framework to analyze duty-cycled throughput achieved by four state-of-the-art MAC schedulers and compares their performance numerically. Section 2.6 experimentally corroborates the analytical results. Section 2.7 draws our conclusions.

2.2 Related Work

Over the past dozen years, extensive research has been conducted to understand how the throughput capacity scales with the number of nodes in wireless ad hoc networks. Gupta and Kumar’s results [34] were extended to account for many factors such as the physical interference model, different spacial distribution of nodes, high/low attenuation regimes [68], and multi-channel networks where nodes may have fewer interfaces than available channels [45]. Although most results suggest that the per node throughput approaches zero as the network size increases, the authors of [51] argue that the key factor that determines whether large ad hoc networks are feasible lies in the locality of traffic. Traditional studies on network capacity assume nodes have a persistent source of power source, however, as low-power ad hoc wireless networks especially wireless sensor networks have started to proliferate, duty cycling has become an obligatory operation to conserve energy and extend lifetime of the
network. To our knowledge, the question of how the network capacity scales in a duty-cycled manner has not been formally studied in previous work.

Regarding duty cycling in multi-hop networks, there have been many attempts at designing energy-efficient routing protocols that minimize latency between source and destination nodes in the context of sleep-wakeup scheduling. For instance, [46] revisited the shortest path problem in asynchronous duty-cycled wireless sensor networks. The authors modeled the time-varying link cost and distance from each node to the sink as periodic functions and made the time-dependent shortest path problem solvable in polynomial-time. Other work studied how to reduce energy usage by introducing energy-efficient sleep-wakeup scheduling [67] or power management scheme [44] for multi-hop sensor networks. Although the proposed schedulers are valuable, their high complexity has translated to limited adoption in the field.

2.3 Duty-Cycled Capacity of Single-Channel Wireless Networks

In this section, we derive bounds on the achievable throughput of random wireless networks given the duty cycle and network density for both multi-hop and single-hop traffic under the protocol model which postulates a geometric condition for successful transmission. We do not consider the physical model in this work, but previous research [68] shows that there is a correspondence between the two models and similar results exist on the scaling behavior of throughput capacity under the physical model.

2.3.1 System Model

Consider a random network where \( n \) nodes are uniformly and independently distributed in a unit square. Each node, \( X_i, i=1,...,n \), has a random destination node
to which it sends data. We consider the case where the transmission range, \( r_n \), as well as the traffic pattern are homogeneous for each node. Each node has a maximum bandwidth of \( W \) bps over a channel and it wakes up on average to communicate for \( \psi \) fraction of time, where \( \psi \in [0, 1] \). Note that \( \psi \) refers to the duty cycle available for each node. We assume that only one channel is available in the network.

According to the protocol model, the transmission from \( X_i \) to \( X_j \) at rate \( W \) bps in a given channel will be successful if the following inequality holds for all other \( X_k \), \( k \neq i, j \) that are concurrently transmitting over the same channel:

\[
|X_k - X_j| \geq (1 + \Delta)|X_i - X_j|.
\]

A circle of radius \((1 + \Delta)|X_i - X_j|\), where \( \Delta > 0 \), quantifies a guard zone required around the receiver within which there is no interference from neighboring nodes transmitting on the same channel at the same time.

### 2.3.2 Capacity of Multi-Hop Traffic

We have the following main result for multi-hop traffic in duty-cycled random wireless networks.

**Theorem 1.** The per node throughput capacity of duty-cycled random wireless networks under the protocol model is

\[
\lambda = \Theta\left(\frac{W}{\Delta \sqrt{n}} \sqrt{\psi} \right) \text{ bps}
\]

until it reaches canonical throughput capacity of \( \Theta\left(\frac{W}{\Delta^2 \sqrt{n \log n}} \right) \) when \( \psi > \frac{K_1}{\Delta^4 \log n} \).

Before we present a proof of the theorem, two comments are in order about the result. First, there is a limit on duty cycle above which the per node throughput stops
growing due to interference and reaches the canonical capacity limit, i.e., \( \Theta\left(\frac{W}{\Delta^2 \sqrt{n \log n}}\right) \) [68]. In other words, with the optimal scheduler there is no capacity gain in operating the network at a duty cycle higher than its limit, which is \( \frac{K}{\Delta^2 \log n} \) as \( n \to \infty \). Second, below the limiting duty cycle, the capacity scales better than \( \Theta\left(\frac{W}{\Delta^2 \sqrt{n \log n}}\right) \) by \( \Theta(\log n) \).

As will be shown in the lower bound proof, duty cycle \( \psi = \Theta\left(\frac{1}{\log n}\right) \), thus the result \( \Theta\left(W \sqrt{\frac{\psi}{n}}\right) \) is equivalent to \( \Theta\left(\frac{W}{\sqrt{\log n}} \cdot \log n\right) \); it is instructive to note that the size of each interference cell in the proof is \( \Theta(\log n) \). Thus the \( \Theta(\log n) \) gain implies that it is feasible to avoid interference and conserve energy at the same time by properly co-scheduling both transmissions and duty cycling.

**Proof.** In the next two sections, we prove Theorem 1 by showing that Eq. (2.2) is both the upper bound and the lower bound of duty cycled capacity.

**Upper Bound on Duty-Cycled Capacity**

In the model, \( r_n \) is the common communication range of nodes in a random network with \( n \) nodes. Since each node needs to communicate with every other node, there is no isolated node in the network. It has been shown by [68] that \( r_n \) should be asymptotically larger than \( \sqrt{\frac{\log n}{n \pi}} \). Moreover, disks of radius \( \frac{\Delta}{2} r_n \) centered at every transmitter should be disjoint for the purpose of collision-free communications. Thus, the area of each disk equals \( \frac{\pi \Delta^2 r_n^2}{4} \). Since at least \( 1/4 \) portion of this area is within the unit square, the maximum number of concurrent transmissions feasible in the network is no more than

\[
\frac{1}{4} \cdot \frac{\pi \Delta^2 r_n^2}{4} = \frac{16 \Delta^2 r_n^2}{4\pi}.
\]

Let \( \tilde{L} \) be the expected distance between two uniformly and independently chosen points within the unit square. Then the expected length from a node to its destination
is $Z = L - o(1)$ since there is always a node within $\Theta(\sqrt{\log n}/n)$ distance from any point (Lemma 5.7 [68]). As a consequence, each packet travels on average $Z/r_n$ hops to reach its destination.

Let each node generate packets at rate $\lambda$, which indicates that the bit rate the network needs so as to accommodate its traffic is at least $n\lambda Z/r_n$, where $0 < Z < 1$. In the optimal schedule, at most $n/2$ senders can be active simultaneously in any given slot. Assume that each sender on average wakes up for $t$ slots out of $T$ slots, which means $\psi = t/T$. The maximum number of potential transmissions in the network in $T$ slots is thus $nt/2$. On the other hand, the number of simultaneous transmissions the network can support is no more than $\frac{16}{\Delta^2 r_n^2 \pi}$. Therefore, the number of achievable transmissions over time period $T$ equals $\frac{16}{\Delta^2 r_n^2 \pi} T$. As long as the number of potential transmissions does not exceed network capacity, that is,

$$\frac{nt}{2} \leq \frac{16}{\Delta^2 r_n^2 \pi}. \quad (2.4)$$

the optimal scheduler may accommodate the traffic. We convert Eq. (2.4) into the following inequality in terms of $\psi$:

$$\frac{n}{2} \psi \leq \frac{16}{\Delta^2 r_n^2 \pi},$$

and

$$\psi \leq \frac{32}{n\Delta^2 r_n^2 \pi} \leq \frac{K_1}{\Delta^2 \log n}. \quad (2.5)$$

In Eq. (2.5), $r_n$ is replaced with a function of $n$ since it is asymptotically larger than $\sqrt{\frac{\log n}{n\pi}}$ for connectivity [68]. As the network capacity is reached, any remaining duty cycle cannot be effectively utilized. Thus, the achievable multi-hop throughput has the following inequality

$$n\lambda \frac{Z}{r_n} \leq W \cdot \min\{\frac{n\psi}{2}, \frac{16}{\Delta^2 r_n^2 \pi}\}. \quad (2.6)$$
Let us first consider the inequation with respect to the second term in the \( \min \) function. Since \( r_n \) is asymptotically larger than \( \sqrt{\log n / n\pi} \), based on Eq. (2.6) we can derive
\[
\lambda \leq \frac{16W}{n\pi\Delta^2r_nZ} \leq \frac{C_1W}{\Delta^2\sqrt{n\log n}}. \tag{2.7}
\]
Second, the first term in the \( \min \) function represents throughput before the maximum capacity (shown in Eq. (2.7)) is reached. By substituting the constraint \( r_n^2 \leq \frac{32}{\Delta^2n\psi} \) obtained from Eq. (2.4), we complete the proof for Theorem 1 with the following
\[
\lambda \leq \frac{W\psi r_n}{2Z} \leq \frac{C_2W}{\Delta} \sqrt{\frac{\psi}{n}}. \tag{2.8}
\]
In summary, the upper bound for the duty-cycled capacity in random wireless networks is as follows. Term \( C_i \) and \( K_i \), where \( i = 1, 2, \ldots \), represent different constants.
\[
\lambda \leq \min\{\frac{C_2W}{\Delta} \sqrt{\frac{\psi}{n}}, \frac{C_1W}{\Delta^2\sqrt{n\log n}}\}. \tag{2.9}
\]

**Lower Bound on Duty-Cycled Capacity**

In the previous section, we established the upper bound on throughput capacity in terms of duty cycle and network density. To prove that the bound in Theorem 1 is tight, we present a scheme that achieves throughput \( \lambda \), where \( \lambda = \frac{C_\psi W}{(1+\Delta)^2} \sqrt{\frac{\pi}{n}} \), for every node when the limiting duty cycle has not been reached.

Consider the scheme presented in Section 5.3 of [68], which achieves throughput \( \lambda = \frac{C_\psi W}{(1+\Delta)^2\sqrt{n\log n}} \) bps for every node in the network to its chosen destination, with probability approaching one as \( n \to \infty \). We only sketch the proof of lower bound for reasons of space. First, the unit square is divided into small cells of such a size that each of them holds at least one but no more than \( O(\log n) \) nodes. Second, these cells are grouped into a finite number of non-interfering sets which can take turns...
in transmission without causing interference. Finally, a simple routing strategy – forwarding a packet from cell to cell “along” the line connecting the originating cell to the destination cell – can fulfill the job.

Their work showed that a transmitting schedule exists such that in every $M^2 = (C_4(1 + \Delta))^2$ time slots, each cell gets one slot to transmit at rate $W$ bps with transmission range $r_n$. By Lemma 5.11 of [68], each cell needs to transmit at a rate less than $\lambda C_5 \sqrt{n \log n}$, with probability approaching one. This can therefore be accommodated by all cells if

$$\lambda C_5 \sqrt{n \log n} \leq \frac{W}{(C_4(1 + \Delta))^2}. \quad (2.10)$$

The required duty cycle in this schedule can be determined as follows. Every cell needs to wake up at a duty cycle of $1/M^2$, or equivalently $1/(C_4(1 + \Delta))^2$, where constraint $C_4 \geq 1$ ensures non-interfering transmission. Given that for any constant $K > 1$, with probability approaching one as $n$ goes to infinity, there are at least one but no more than $Ke \log n$ nodes in a cell and only one node is allowed to transmit in every cell each time. The duty cycle for every node to transmit is thus

$$\psi = \frac{1}{M^2 Ke \log n} = \frac{1}{C_4^2(1 + \Delta)^2 Ke \log n}. \quad (2.11)$$

By combining Eq. (2.10) and (2.11), the throughput capacity can be converted in terms of duty cycle and network density as follows.

$$\lambda C_5 \sqrt{n} \leq \frac{W}{C_4(1 + \Delta)} \cdot \frac{1}{C_4(1 + \Delta) \sqrt{\log n}}$$

$$\lambda C_5 \sqrt{n} \leq \frac{C_6 W}{(1 + \Delta) \sqrt{\psi}},$$

$$\lambda \leq \frac{C_7 W}{(1 + \Delta) \sqrt{\psi/n}}, \quad \text{where} \quad \psi \leq \frac{K_1}{\Delta^2 \log n}. \quad (2.12)$$
Note that achievability of the second term in \( \min \) of Eq. (2.9) is already proven in [68]. Thus the derivation above proves that the lower bound of duty cycled capacity is identical to the upper bound, thereby proving Theorem 1.

2.3.3 Capacity of Single-Hop Traffic

A common observation regarding throughput capacity is that the throughput available to each node approaches zero as the number of nodes increases. Nevertheless, [51] shows that the locality of traffic is the key factor in deciding whether large ad hoc networks are feasible. This work corroborates the fact that local traffic is preferable in large-scale ad hoc networks. In this section we specialize our analysis for single-hop traffic and present the corresponding capacity for both general and clique networks with two corollaries.

For the case of single-hop traffic, \( Z/r_n \) has a constant value of 1, which leads to the following corollary on the duty-cycled capacity in general networks:

**Corollary 1.** The per node throughput capacity of a duty-cycled random wireless network with single-hop traffic is

\[
\lambda = \Theta(W\psi) \text{ bps}
\]

(2.13)

until it reaches its network capacity of \( \Theta(\frac{W}{\Delta^2 \log n}) \) when \( \psi > \frac{K_1}{\Delta^2 \log n} \).

**Proof.** We adjust the capacity inequality Eq. (2.6) by substituting \( Z/r_n \) by 1 as follows.

\[
n\lambda \cdot 1 \leq W \cdot \min\{\frac{n\psi}{2}, \frac{16}{\Delta^2 r_n^2 \pi}\}
\]

(2.14)
Hence,

\[ \lambda \leq W \cdot \min\{ \frac{\psi}{2} \cdot \frac{16}{\Delta^2 \log n} \}. \tag{2.15} \]

Note that the network capacity increases from \( \Theta\left( \frac{W}{\Delta^2 \sqrt{n \log n}} \right) \) to \( \Theta\left( \frac{W}{\Delta^2 \log n} \right) \) under single-hop traffic. With respect to the duty-cycled capacity, \( \Theta(\log n) \) throughput gain is achieved against \( \Theta\left( \frac{W \psi}{\log n} \right) \).

For the special case of clique networks, the maximum number of simultaneous transmissions reduces from \( \frac{16}{\Delta^2 r^2 \pi} \) to 1, which leads to our next corollary:

**Corollary 2.** The per node throughput capacity of a duty-cycled clique network with single-hop traffic is

\[ \lambda = \Theta(W \psi) \text{ bps} \tag{2.16} \]

until it reaches its network capacity of \( \Theta\left( \frac{W}{n} \right) \) when \( \psi > \frac{2}{n} \).

**Proof.** We degenerate Eq. (2.14) further by replacing \( \frac{16}{\Delta^2 r^2 \pi} \) with 1:

\[ n\lambda \cdot 1 \leq W \cdot \min\{ \frac{n\psi}{2}, 1 \}, \]

\[ \lambda \leq W \cdot \min\{ \frac{\psi}{2}, \frac{1}{n} \}. \tag{2.17} \]

Notably, below the limiting duty cycle, per node throughput grows linearly with \( \psi \), which equals \( \Theta(W \psi) \), as opposed to \( \Theta(\frac{W \psi}{n}) \). The \( \Theta(n) \) throughput gain (instead of \( \Theta(\log n) \)) is because the interference size of a clique network is \( \Theta(n) \).
2.4 Duty-Cycled Capacity of Multi-Channel Wireless Networks

Beyond single-channel wireless networks, this section investigates the duty-cycled capacity of a random network with multiple channels. We use the term “channel” to refer to some frequency band with a specified bandwidth. Assume that there are $c$ channels in total, and every node is equipped with $m$ interfaces. Each interface is capable of transmitting or receiving data on any one channel at a given time. And each node is allowed to switch its interface from one channel to another instantly; that is, we assume that there is no delay in switching an interface from one channel to another.

According to Gupta and Kumar [34], the capacity per node with $c$ channels and $m = c$ interfaces per node scales as $\Theta(Wc\sqrt{\frac{1}{n\log n}})$, where the aggregate data rate achievable by using all $c$ channels is $Wc$ bps. Previous research on multi-channel capacity [45] shows that the capacity of multi-channel networks exhibits different bounds which are dependent on the ratio between $c$ and $m$. When the number of interfaces per node is less than the number of channels, there is a degradation in the network capacity in many scenarios. However, one exception is for a random network with up to $O(\log n)$ channels, wherein the capacity per node remains at the Gupta and Kumar bound, independent of the number of interfaces available at each node.

In the model, we assume that the number of available channels, $c$, satisfies the constraint $1 \leq c \leq O(\log n)$. Furthermore, The $c$-channel-single-interface model is adopted to reflect the current technology of wireless platforms. We have the following result for duty-cycled random wireless networks with multiple channels.
Theorem 2. The per node throughput capacity of duty-cycled random wireless networks with $c$ channels and single interface under the protocol model is

$$\lambda = \Theta\left(\frac{W}{\Delta V} \sqrt{\frac{\psi c}{n}} \right) \text{ bps} \quad (2.18)$$

until it reaches canonical throughput capacity of $\Theta\left(\frac{Wc}{\Delta^2 \sqrt{n \log n}} \right)$ when $\psi > \frac{K_1 c}{\Delta^2 \log n}$.

This result implies that the capacity of multi-channel random networks increases with $\sqrt{c}$ before reaching the limit of duty cycle. The limiting duty cycle increases by a factor of $c$ as well.

Proof. In the following two sections, we prove Theorem 2 by showing that Eq. (2.18) is both the upper bound and the lower bound of duty cycled capacity for multi-channel networks.

2.4.1 Upper Bound on Duty-Cycled Capacity

Given $c$ available channels, the maximum number of concurrent transmissions in the network increases by $c$ times, which equals $\frac{16c}{\Delta^2 r_n^2 \pi}$. Hence, Eq. (2.4) and (2.5) can be extended as follows.

$$\frac{nt}{2} \leq 16Tc \frac{2}{\Delta^2 r_n^2 \pi}$$

$$n \psi \leq \frac{16c}{\Delta^2 r_n^2 \pi}$$

$$\psi \leq \frac{32c}{n \Delta^2 r_n^2 \pi} \leq \frac{K_1 c}{\Delta^2 \log n}. \quad (2.19)$$

Accordingly, the network capacity is divided into two regions:

$$n\lambda \frac{Z}{r_n} \leq W \cdot \min\left\{ \frac{n\psi}{2}, \frac{16c}{\Delta^2 r_n^2 \pi} \right\}. \quad (2.20)$$
Similarly to previous steps in Eq. (2.7) and (2.8), we may obtain the following upper bound of the duty-cycled capacity. The derivation steps are omitted here due to space limitation.

\[
\lambda \leq \min \left\{ \frac{C_8 W}{\Delta} \sqrt{\frac{\psi c}{n}}, \frac{C_9 W c}{\Delta^2 \sqrt{n \log n}} \right\}.
\] (2.21)

### 2.4.2 Lower Bound on Duty-Cycled Capacity

The lower bound is established by constructing an appropriate routing scheme and transmission schedule. We examine the construction of routing and transmission scheme in [45] and show that the upper bound in Eq. (2.21) is achievable by analyzing the desired duty cycle of the proposed schedule.

The unit square network is divided into square cells, each of area \( a(n) \). The size of \( a(n) \) is carefully chosen to meet multiple constraints [45]. By construction, we have \( a(n) \geq \frac{100 \log n}{n} \). The transmission schedule is built using a two-step process. In the first step, transmissions are scheduled in “edge-color” slots such that at every node during any edge-color slot, at most one transmission or reception is scheduled. Since edge coloring ensures that at a vertex, all edges connected to the vertex use different colors, each node will have at most one transmission or reception scheduled in any edge-color slot. In the second step, each edge-color is divided into \( \lceil \frac{K_2 n a(n)}{c} \rceil \) “mini-slot” for some constant \( K_2 \). Each node is assigned a mini-slot for transmission or reception without interfering with others via vertex coloring. As a consequence, the average duty cycle of a node under this schedule is

\[
\psi = \frac{1}{\lceil \frac{K_2 n a(n)}{c} \rceil} = \frac{c}{K_2 n a(n)}.
\] (2.22)
Moreover, the achievable rate for each flow is [45]

\[ \lambda = \Omega\left(\frac{Wc}{K_2n\sqrt{a(n)}}\right). \]  

(2.23)

By substituting \( \sqrt{a(n)} \) with a function of \( \psi \), we may obtain the capacity as

\[
\lambda = \Omega\left(\frac{Wc}{K_2n\sqrt{\psi_c}}\right) = \Omega\left(W\sqrt{\frac{\psi_c}{K_2n}}\right) \\
= \Omega\left(\frac{W}{(2 + \Delta)}\sqrt{\frac{\psi_c}{n}}\right). 
\]

(2.24)

Note that the constant \( K_2 \) chosen in [45] has the value \( C_{10}(2 + \Delta)^2 \), where \( C_{10} \geq 1 \).

Since in the construction we have \( a(n) \geq \frac{100\log n}{n} \), the scheduled duty cycle \( \psi \) satisfies the following restriction

\[
\psi \leq \frac{c}{100K_2\log n} \leq \frac{c}{100\Delta^2\log n} \leq \frac{K_1c}{\Delta^2\log n}. 
\]

(2.25)

Therefore, the duty cycled capacity for random networks with \( c \) channels and single interface have identical upper bound and lower bound, which concludes the proof for Theorem 2.

\[ \Box \]

### 2.5 Duty-Cycled Capacity Achieved by Single-Channel MACs

To study the gap between the duty-cycled capacity achieved by state-of-the-art MAC protocols and the optimal scheduler, we abstract existing state-of-the-art MAC protocols based on time synchrony and the scheduling centricity, and define four schedulers each of which represents a popular class of CSMA MAC protocols in low-power wireless sensor networks. TDMA protocols belong to a different school of MAC scheduling, yet maintaining of its schedule involves high control overhead. Thus, they are not so widely used in low-power wireless networks as CSMA MAC protocols, and are excluded from the comparison in this work.
We first introduce the rationale of MAC classification. Accordingly, we divide ten well-known MAC protocols into four classes. Then, we compare protocols within the same class and choose one representative for further evaluation. Fundamentally, MAC protocols manage local traffic. Therefore, we consider the duty-cycled capacity achieved for single-hop traffic, where nodes are paired for communication. We propose a general framework that can be applied to all four schedulers. Achievable throughput of the different schedulers are analyzed individually in terms of the available duty cycle and network density. Lastly, we compare numerically duty-cycled capacities achieved by these MAC schedulers.

2.5.1 Rendezvous Elements

We focus on two rendezvous elements that essentially determine the scheduling of communications: coordination centricity and synchrony. As we will illustrate in the next subsection, a nontrivial number of MAC scheduling protocols can be categorized according to their choices in these two elements.

A basic function of duty-cycled MAC protocol is to ensure rendezvous since connectivity between nodes becomes time-dependent in the presence of sleep-wakeup cycling. Since radios are not always active, coordination centricity decides whether the sender or the receiver is responsible for initiating their communication. In sender-centric protocols, the sender chooses the time to transmit and the receiver has to be active when the data communication happens; whereas in receiver-centric protocols the sender transmits according to the receiver’s wakeup schedule, which means the sender has to transmit, when the receiver is active. (The alternative that the sender
and the receiver both share the responsibility is typically eschewed given its higher overhead."

Regarding the level of synchrony, the choices are asynchronous, locally synchronous, or globally synchronous operation. In *asynchronous* protocols, nodes act independently according to their local time. Since nodes do not share a common notion of time, schemes such as a long preamble or idle listening have to be used to guarantee rendezvous. On the other hand, locally (respectively, globally) synchronous protocols require that neighboring (respectively, all) nodes share the same notion of time and are thus capable of coordinating accordingly.

### 2.5.2 Taxonomy of MAC Protocols

Over the last decade, extensive research has spawned a rich body of duty-cycled MAC protocols [58, 18, 54, 70, 63, 35, 61, 21, 28], in which the radio is switched on and off periodically according to the MAC schedule. To provide a better understanding of different MAC schedules, Table 2.1 summarizes ten well-known protocols into four classes in terms of the two rendezvous elements. The exemplar we chose for each class is in bold font.

<table>
<thead>
<tr>
<th>Syn.</th>
<th>Receiver-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCP-MAC [69], DW-MAC [62], T-MAC [63], S-MAC [70]</td>
<td>O-MAC [21], WiseMAC [28]</td>
</tr>
<tr>
<td>Asyn.</td>
<td>BoX-MAC [54], X-MAC [18], B-MAC [58]</td>
</tr>
</tbody>
</table>

Table 2.1: MAC protocols of different classes
Due to the extensive number and diversity of existing MAC protocols, it is infeasible to include all proposed MACs in the comparison. Instead, we focus on duty-cycled CSMA-based MAC protocols that have predominantly been adopted in WSN deployments. Thus Table 2.1 excludes non-duty-cycled MACs [40, 22, 32], cross-layer protocols that are tailored to address specific application settings such as for ultra-low traffic [19, 55], and Time Division Multiple Access (TDMA) protocols that are not widely used in WSN applications due to the high control overhead.

**Inner-class comparison**

Before we perform comparative study on different MAC classes, inner-class MAC comparison is conducted to review minor differences within each class and to choose one representative MAC per class for further study. These exemplars were shown to be the most energy efficient in each class, which are SCP-MAC, BoX-MAC, O-MAC, and RI-MAC, respectively.

First, in the **sender-centric synchronous** class, S-MAC [70] introduced the idea of duty cycling radios to WSNs for energy conservation. Nodes synchronize to wake up periodically, to transmit or receive data, and then return to sleep. Senders choose the time to transmit, using an RTS/CTS scheme to coordinate with neighbors and then transmit packets. T-MAC [63] minimized idle listening in S-MAC by listening to the channel for only a short window after the synchronization phase and returning to sleep if no data is received during this window. DW-MAC [62] further reduces the latency due to constant sleep intervals by splitting the contention and data communication phase such that nodes may wake up more often for scheduled transmissions. Likewise, SCP-MAC [69] allows nodes to adapt channel polling rate to incoming traffic rates. In addition, senders perform two-phase CSMA contention instead of the costly RTS/CTS
to lower the probability of collision. The MAC comparison in [47] demonstrates that SCP-MAC outperforms its predecessors such as S-MAC and T-MAC in terms of both energy efficiency and latency.

Second, in the **sender-centric asynchronous** class, B-MAC [58] introduced the mechanism of Low Power Listening (LPL). Receivers independently and periodically poll the channel for activity at low energy cost; senders precede data transmission with a preamble that is slightly longer than the wakeup interval of the receiver. With the preamble, a sender is assured that the receiver detects the preamble upon wakeup and remains awake in order to receive data. Later, X-MAC [18] replaced the long preamble of B-MAC with a series of short probes, each of which contains the destination address, allowing non-intended receivers to return to sleep without overhearing the entire preamble. In a more recent enhancement, BoX-MAC [54] enabled not only early preamble termination as in X-MAC by separating the consecutive probes with gaps long enough for early acknowledgements from the destined receiver, but also the use of data packets as probes to further reduce the overhead of transmitting preambles.

Third, in the **receiver-centric (locally) synchronous** class, both WiseMAC [28] and O-MAC [21] allowed senders to locally track their receiver’s regular wakeup schedule. Via local synchronization, senders with pending data thus wake up just before their intended receiver and contend to transmit. WiseMAC chose to not send time sync packets explicitly, instead, it piggybacks time information with data packets. To compensate for the possible loss of synchrony when data traffic is rare, WiseMAC has to use preamble, the length of which depends on the level of clock drift. In the more recent O-MAC protocol, receivers select wakeup slots according to a seed-based
pseudo-random number generator to stagger their wakeup times. Senders obtain the seed from the receiver and can thus thereafter predict the receiver’s wakeup time. Local synchronization is maintained by piggybacking information, or independent synchronization phase in case the data rate is extremely low.

Lastly, in the **receiver-centric asynchronous** class, RI-MAC [61] avoided channel-intensive preambles by letting receivers independently and periodically broadcast short beacons to invite potential transmissions. To rendezvous with the receiver, a sender turns on radio whenever traffic arrives and silently waits for beacon from the intended receiver to transmit data. Thus, the time a sender occupies the wireless medium for rendezvous is reduced significantly compared to the sender-centric asynchronous class. In summary, asynchrony is easier to implement but necessitates the use of either a long preamble or idle listening as a means of rendezvous.

### 2.5.3 Framework for MAC Schedulers

We assume that time is divided into consecutive slots and at most one packet may be transmitted during each slot. The duty cycle $\psi$ is thus interpreted as the ratio of the number of active (awake) slots to the total number of slots that are considered. Note that there are a number of ways to implement a duty cycle $\psi$ at the MAC layer, however, this abstract framework does not focus on the implementation of $\psi$. Instead, it assumes that MAC schedulers adopt a similar scheme in instantiating their duty cycles.

The optimal MAC scheduler guarantees that only one node transmits in the interference region and no energy is wasted in scheduling, thus the delivery ratio per
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W$</td>
<td>maximum transmission rate (bps)</td>
<td>250$K$</td>
</tr>
<tr>
<td>$w$</td>
<td>contention window size (# of timeslices)</td>
<td>16</td>
</tr>
<tr>
<td>$u$</td>
<td>beacon length relative to data length</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 2.2: Model parameters

node equals its duty cycle $\psi$. In contrast to the optimal scheduler, CSMA MAC protocols only guarantee a probability of successful communication in each slot, which is represented by $\tau(\psi)$, where $\tau(\psi) \in [0, 1]$. Thus, the expected number of successful transmissions for $n/2$ senders is $n \cdot \tau(\psi)/2$. Accordingly, by substituting $\psi$ with $\tau(\psi)$ in Eq. (2.15) and (2.17), the throughput capacity becomes

$$\lambda \leq W \cdot \min \left\{ \frac{\tau(\psi)}{2}, C(n) \right\}, \quad (2.26)$$

where

$$C(n) = \begin{cases} 
16/\left(\Delta^2 \log n \right), & \text{general network} \\
1/n, & \text{clique network}.
\end{cases}$$

Next, we present a framework for calculating $\tau(\psi)$ for all four schedulers, which subsumes four representative MACs. Typically, when a node attempts to transmit a packet, it first randomly selects one out of $w$ timeslices in the contention window and then monitors the channel up to that timeslice to ensure that no other transmission occurs in the communication range of the node. If any transmission is detected before the chosen timeslice, the sender aborts its transmission attempt; otherwise, it immediately starts the data transmission after the timeslice. The probability of selecting any timeslice is $1/w$. Let $\hat{\epsilon}$ represent the expected number of contenders with respect to a sender. The value of $\hat{\epsilon}$ is MAC dependent, which will be instantiated in each MAC model. The probability for a sender to uniquely access the channel in its
communication range is denoted by $p_s$, where

$$p_s = \frac{1}{w} (1 - \frac{1}{w})^\epsilon + \frac{1}{w} (1 - \frac{2}{w})^\epsilon + \ldots + \frac{1}{w} \cdot (\frac{1}{w})^\epsilon,$$

$$= \frac{1}{w} \sum_{i=1}^{w} (1 - \frac{i}{w})^\epsilon. \quad (2.27)$$

Let the number of potential contenders within a node’s communication range be $\epsilon$ and the number of potential contenders in the interference range of a node be $\eta$. Typically, we have $\eta \geq \epsilon$. A transmission is guaranteed to succeed if the sender secures the channel exclusively and no other senders access the channel within the interference range of the receiver. Hence, the total probability for a successful delivery in any slot is:

$$\tau(\psi) = p_d \cdot p_s (1 - p_d \cdot p_s)^{\eta - \epsilon}, \quad (2.28)$$

where $p_d$ denotes the probability that a node attempts to send a packet in any slot. Term $p_d \cdot p_s$ is the probability that a sender wins the channel and transmits a packet to its destination. The value of $p_d$ will be discussed at the end of this section. To ensure that no transmission comes from nodes in the interference range, $\eta - \epsilon$ nodes have to remain silent during this slot. According to the system model in Section 2.3, the communication range and interference range for each node are disks of radius $r_n$ and $(1 + \Delta) r_n$, respectively. At most half of nodes serve as senders, thus $\epsilon$ and $\eta$ in the unit square network model are respectively

$$\epsilon = \frac{\pi r_n^2 n}{2}, \quad \text{and} \quad \eta = \frac{\pi ((\Delta + 1)^2 r_n^2 n}{2}. \quad (2.29)$$

For ease of experimental validation, let us consider a clique network, where communication and interference areas are the same. These parameters simply become
Duty Cycle Constraint

<table>
<thead>
<tr>
<th>MAC</th>
<th>Duty Cycle Constraint</th>
<th>( \epsilon ) (( \epsilon = n/2 ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCP-MAC</td>
<td>( p_t \psi_r + \psi_r = 2\psi )</td>
<td>( p_t \epsilon )</td>
</tr>
<tr>
<td>O-MAC</td>
<td>( p_t \psi_r + \psi_r = 2\psi )</td>
<td>( p_t \psi_r \epsilon )</td>
</tr>
<tr>
<td>BoX-MAC</td>
<td>( \left( \frac{1}{2\psi_r} + 1 \right) \cdot p_t \psi_r + \psi_r = 2\psi )</td>
<td>( \left( \frac{p_t}{2} + p_t \psi_r \right) \epsilon )</td>
</tr>
<tr>
<td>RI-MAC</td>
<td>( \left( \frac{1}{2\psi_r} + 1 \right) \cdot p_t \psi_r + \psi_r = 2\psi )</td>
<td>( (p_t + u) \psi_r \epsilon )</td>
</tr>
</tbody>
</table>

Table 2.3: MAC dependent parameters for four MAC schedulers

\( \epsilon = \eta = n/2 \). The probability of successful delivery is accordingly

\[
\tau(\psi) = p_d \cdot p_s. \tag{2.30}
\]

Since every sender has a dedicated receiver that is one hop away, the total duty cycle of any sender-receiver pair equals \( 2\psi \), out of which the receiver spends \( \psi_r \), \( 0 \leq \psi_r \leq 1 \), in receiving mode to account for the time over which a node waking up, polls the channel, possibly receives a packet, and goes to sleep. A corresponding sender may send data with probability \( p_t \) when the receiver is awake, leading to the equation

\( p_d = p_t \cdot \psi_r \). The relationship between variables \( p_t \) and \( \psi_r \) is MAC dependent, and will be instantiated for each MAC model in the following section.

2.5.4 Case Study of MAC Schedulers

In this section we elaborate on how to instantiate parameters in the abstract framework consisting of equations from Eq. (2.26) to (2.30) for each scheduler.

SCP-MAC: Sender-Centric Synchronous

In SCP-MAC [69], all nodes in a region wake up simultaneously in each frame. The wakeup (or channel polling) interval is also known as the frame length. Since nodes poll the channel for activity in an aligned fashion, senders contend to transmit data just before receivers wake up. Such synchronized polling not only reduces the
energy cost in rendezvous, but also improves channel utilization during the wakeup slots.

Given a receiver spends duty cycle $\psi_r$ in receiving mode, the corresponding sender would transmit for a duty cycle of $p_t \cdot \psi_r$, where $0 \leq p_t \leq 1$. Thus, the total $2\psi$ is distributed as

$$p_t \cdot \psi_r + \psi_r = 2\psi,$$

$$\psi_r = \frac{2\psi}{1 + p_t}, \quad (2.31)$$

where the constraint, $0 \leq \psi_r \leq 1$, has to be satisfied. Due to the global synchrony feature, the expected number of active contenders when a node attempts to transmit equals the product of the number of potential senders in communication range, $\epsilon$, and the probability that a sender intends to transmit. Thus,

$$\hat{\epsilon} = p_t \cdot \epsilon. \quad (2.32)$$

Given each $\psi$ over the range $[0, 1]$, we may determine the optimal value of $\tau(\psi)$ by varying $p_t$ in range of $[0, 1]$.

**O-MAC: Receiver-Centric Synchronous**

O-MAC [21] is receiver-centric and locally synchronous: receivers use seed-based pseudo-random wakeup schedules to wake up independently and periodically. The seeds for their schedules are broadcast to neighbors in order to maintain local synchrony. Senders with pending data thus wake up just before their intended receiver does. The sharing of the seeds makes the use of preambles or probes unnecessary.

Since sender-receiver pair is synchronized, the utilization of duty cycle follows the same equality in Eq. (2.31). The expected number of interferers with respect to
a node, however, reduces by a fraction of $\psi_r$ due to independent wakeup times of receivers. Thus,

$$\hat{\epsilon} = p_t \cdot \psi_r \cdot \epsilon.$$  \hspace{1cm} (2.33)

**BoX-MAC: Sender-Centric Asynchronous**

The more recent BoX-MAC [54] adopts the Low Power Listening (LPL) mechanism, in which receivers independently and periodically poll the channel for activity using low power. Each sender wakes up its receiver by sending it duplicate back-to-back data packets until it receives an acknowledgement from the intended receiver.

Since a receiver spends $\psi_r$ of time in receiving mode, the frame length is approximately $1/\psi_r$ slots. The expected number of slots for preamble transmission until the receiver waking up is thus half of the frame length, i.e., $\frac{1}{2\psi_r}$. Each time a sender transmits to its receiver, it spends $(\frac{1}{2\psi_r} + 1)$ time in sending preambles and data, thus the total duty cycle of $2\psi$ is divided into

$$\frac{1}{2\psi_r} + 1 \cdot p_t \cdot \psi_r + \psi_r = 2\psi,$$

$$\psi_r = \frac{4\psi - p_t}{2(p_t + 1)},$$ \hspace{1cm} (2.34)

where the constraint, $0 \leq \psi_r \leq 1$, has to be satisfied. The expected number of contenders becomes Eq. (2.35) since the transmission of preamble increases channel occupancy rate approximately by a factor of $(\frac{1}{2\psi_r} + 1)$.

$$\hat{\epsilon} = (\frac{1}{2\psi_r} + 1) \cdot p_t \cdot \psi_r \cdot \epsilon = (\frac{p_t}{2} + p_t \psi_r)\epsilon.$$ \hspace{1cm} (2.35)

**RI-MAC: Receiver-Centric Asynchronous**

In RI-MAC [61], nodes independently and periodically wake up to broadcast a beacon to their neighbors. Once senders have data to send they keep their radio active
in receiving mode and contend for the channel upon receiving a beacon from their intended receiver. However, the energy consumption in receive and transmit mode are approximately the same on current radios [6]. Thus, the rendezvous scheme is analogous to BoX-MAC as in Eq. (2.34) except that senders wait quietly for the short beacon from the correspondent receiver. We count the constant length of receiver beacon for $u$ slot, where $0 < u \leq 1$. Accordingly, $\hat{\epsilon}$ now becomes

$$\hat{\epsilon} = (p_t + u) \cdot \psi_r \cdot \epsilon.$$  \hspace{1cm} (2.36)

2.5.5 Comparison of MAC Throughput

Based on the instantiated framework, we evaluate capacities of exemplar MAC protocols across a rich range of duty cycles and network densities for a clique network. The throughput capacity achieved by each MAC protocol at a given duty cycle and network density is determined by optimizing $\lambda$ in Eq. (2.26) over variable $p_t$ subject to the duty cycle constraints. We use the representative values summarized
in Table 2.2 for the constant parameters and the key constraints listed in Table 3.2 for the optimization. Fig. 2.1 plots the optimal scheduler and MAC equations using MATLAB for different schedulers with duty cycles from 0 to 1 and network sizes from 0 to 30 nodes.

Fig. 2.1 shows that the per node throughput capacity in clique networks decreases gradually as the number of nodes grows. Given a network density, the per node throughput of the optimal scheduler increases with duty cycle linearly until it reaches thresholds of $2/n$. We observe that of the four protocols, O-MAC approximates the optimal scheduler best. It achieves the scaling gain by spreading receiver wakeup times to random slots, which makes communications more uniform and independent. Furthermore, it avoids generating unnecessary communication overhead (such as preambles or beacons) by synchronizing nodes locally. When duty cycle increases, i.e., nodes wake up more frequently for data transmission, inter-flow contention becomes nontrivial, thus resulting in a larger gap in performance between O-MAC and the optimal scheduler.

The other synchronous protocol SCP-MAC outperforms asynchronous MACs in most configurations since the synchrony in SCP-MAC substantially reduces the overhead of rendezvous. Unlike O-MAC, SCP-MAC does not attempt to spread communications into mutually exclusive times; instead, it let receivers wake up simultaneously for communications, which increases contention from senders and results in inferior throughput to O-MAC. As we see in Fig. 2.1, as inter-flow contention grows with density, RI-MAC takes over SCP-MAC in performance; however, the throughput of RI-MAC degrades quickly at larger network sizes and higher duty cycles. The advantage of using beacons in RI-MAC thus turns out to be limited to relatively low
traffic since the number of failures in sending out beacons grows quickly as traffic rate increases.

Although asynchronous schedulers such as RI-MAC and BoX-MAC randomly spread wakeup times of receivers, they suffer from their costly rendezvous mechanisms, especially when traffic is low. For instance, BoX-MAC achieves the lowest throughput at low duty cycles, but it starts to catch up with others as duty cycle increases since its rendezvous overhead (preamble length) reduces as nodes wake up more often. At full duty cycle, all MAC protocols converge to a pure CSMA scheme except for RI-MAC whose use of beacons becomes a major constraint.

2.6 Experimental Validation

In this section, we validate the soundness of MAC models for three protocols across a wide range of duty cycles and network densities. All experiments were performed on the TelosB platform in a clique network selected from the large-scale Kansei testbed [10]. Our experiments evaluated existing implementations of O-MAC and BoX-MAC\textsuperscript{1} in TinyOS-2.1 and of RI-MAC in TinyOS-2.0.2. The original SCP-MAC implementation was for Mica2 motes in TinyOS-1.x. A recent version of SCP-MAC in MLA [42] has not yet implemented the overhearing avoidance feature described in the paper, which would negatively impact the efficiency of SCP-MAC. Thus we could not include SCP-MAC in the experiments.

\textsuperscript{1}The BoX-MAC-2 version we tested is included as the default MAC protocol for TinyOS-2.1
2.6.1 Experiment Configuration

To monitor the duty cycle of a node, we implemented a software module in the TinyOS CC2420 driver that is used by all three MAC protocols to measure the cumulative active period of the radio. The radio is considered active upon detecting the oscillator stabilization signal and inactive upon detecting the radio stop signal. Each packet was configured to be 40 bytes long, including the 802.15.4 header, and each byte takes approximately 32 $\mu$s to transmit.

We configured each MAC as follows: packets were buffered in a FIFO queue and retransmission was enabled for up to 5 times per packet. Sequence numbers were attached to data packets to filter out duplicate packets. A set of frame lengths were tested to seek the best performance corresponding to different incoming traffic rates. To decouple the sender duty cycle from the receiver’s and since traffic is one-way in the network, sender frame length was set to a large constant, namely, 100 seconds. Each experiment was conducted for a duration of 15 minutes. The measurement of the performance began when all flows had started transmitting data.

We corroborated the modeling framework by conducting experiments across a rich set of selected network configurations. Since the duty cycle of a node should not be controlled directly by MAC users, we controlled other available parameters instead: the traffic rate and frame length, and monitored the corresponding duty cycle. For each MAC protocol, four network densities (2, 4, 6, 8 nodes), five traffic rates (1, 4, 8, 16, and 32 packets per second), and six different frame lengths (25, 50, 100, 200, 500, and 1000 ms) were instantiated, leading to 120 experiments per MAC protocol. Importantly, for a given network density and traffic rate, we selected the best throughput of each protocol by searching over the six frame lengths.
Figure 2.2: Experimental comparison of per node throughput at diverse network configurations in a clique.

2.6.2 MAC Model Validation

Fig. 2.2 plots the measured throughput of three MAC protocols with respect to the consumed duty cycle and network density. Since the measured duty cycle is not uniformly distributed, several points in the figure were interpolated using the average value of the neighborhood. We observe that the experimental results on the duty-cycled capacity for the three MAC protocols are in line with the trends shown in Fig. 2.1. The correlation coefficient between experimental and analytical results across all evaluated configurations are 0.95 for BoX-MAC and O-MAC, and 0.85 for RI-MAC, respectively. During experiments, we noted a significant number of failures in sending out beacons in RI-MAC at high traffic rates, which led to much lower throughput than expected.

Shown in Fig. 2.2, O-MAC achieves the highest duty-cycled capacity across the tested configurations. At low duty cycles, the performance of RI-MAC is superior to BoX-MAC; since receivers wake up rarely at low traffic rates, BoX-MAC suffers
from higher channel occupancy caused by transmitting long preambles. As duty cycle and traffic rate increase, BoX-MAC tends to use shorter preambles that reduce rendezvous overhead, while RI-MAC starts to experience more frequent failures in beacon transmission. Thus the throughput of RI-MAC degrades substantially at high duty cycles.

Although the major radio activities have been modeled by the modeling framework, certain engineering overheads still exist in protocol implementations. In addition, each packet was 40 bytes in experiment which did not fully utilize the channel. Thus, the experimental throughput is generally much lower than analytical results. Nevertheless, the duty-cycled capacity trends of examined MAC protocols are validated by the high correlation coefficient.

2.7 Conclusions

We have shown that capacity gains resulting from duty cycling, by spreading potential interference over time, can provide asymptotic scalability subject to certain constraints on duty cycle relative to the number of nodes. For random wireless deployments, with multi-hop traffic, all nodes in interference cells, which have log $n$ size, can be spread in time to achieve a $\Theta(\log n)$ throughput gain. And with single-hop traffic, all interfering sources in a locality of size $n$ can be spread over time to achieve a $\Theta(n)$ throughput gain.

Armed with this understanding for local traffics, we showed how different classes of CSMA MAC protocols are systematically analyzed by first developing a parameterized framework that captures the essence of CSMA and by then instantiating each specific scheduling class by properly configuring a small set of parameters. This enabled us
to theoretically compare the duty-cycled capacity of existing MACs and the optimal MAC, and to corroborate our analysis via extensive experiments across a variety of duty cycles and network densities.

Our results indicate that the class of receiver-centric synchronous MAC protocol (such as O-MAC) approximates the optimal scheduler best compared to other MAC schedulers. Staggering receiver wakeup times and synchronizing nodes locally not only avoid creating unnecessary communication overhead, they also make communications more uniform and independent, which is preferable in low-power wireless networks.
Chapter 3: ENERGY EFFICIENCY ACHIEVED BY DUTY-CYCLED CSMA-BASED MAC PROTOCOLS

WSN applications vary widely in their traffic and node density patterns. Conventional wisdom says that a MAC protocol that performs well for one application can perform poorly for another. Perhaps as a result of this view, a large number of MAC protocols have been proposed, often with specific performance metrics in mind; many of the widely used ones have focused on the low duty cycle case. In this chapter, we study how the choice of the MAC protocol as well as the configuration of its parameters impacts performance for diverse traffic rates and node densities.

Specifically, we adopt the classification of duty-cycled CSMA-based MAC protocols in Chapter 2 and introduce a framework for performance modeling of each MAC class as a function of key protocol parameters. We use this framework to analyze various performance metrics comprehensively across the configuration space of the protocols; extensive experimentation corroborates our analysis. Our results serve not only as a basis for comparing protocols, they also yield insight into how to adapt MACs to changing traffics in a distributed way. The potential for extending the framework to analyze richer application scenarios is also investigated and discussed. A surprising finding of our comparative evaluation is that one class of MAC protocols
consistently achieves the best or close to the best performance for various metrics across much of the configuration space.

3.1 Introduction

Over a decade of extensive research, many duty-cycled MAC protocols have been proposed to maximize performance in terms of one or more metrics such as throughput, latency, energy consumption, and memory. How well each MAC performs depends on the specific “configuration” it is instantiated in. The configuration includes not only the network configuration, —the set of network parameters such as network density, traffic rate, and node topology— but also the protocol configuration, —the set of protocol parameters that determine how a protocol communicates and allocates resources given a network configuration.

Extant WSNs often operate in a broad range of traffic rates [13, 64, 36]. However, the design and evaluation of MAC protocols has typically focused on relatively narrow portions of the configuration space. Table 3.1 summarizes some performance comparisons reported in the literature. With one exception, all of these comparisons consider relatively low traffic loads. Moreover, some of them [17, 38, 35, 42] do not attempt to select, for each protocol being compared, protocol configurations (parameters) that respectively optimize its performance. In other words, the performance characterization of state-of-the-art MAC protocols is incomplete and insufficient for comparing the performance of MAC protocols across the configuration space and for addressing the choices that designers, deployers, and managers need to make.

In this chapter, we take a step towards the characterization of MAC performance across the configuration space. We classify the rather large set of CSMA-based MACs
<table>
<thead>
<tr>
<th>Metric</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>Always-on CSMA &gt; WiseMAC [38], BoX-MAC &gt; X-MAC [54], B-MAC &gt; SCP-MAC &gt; Crankshaft (high load) [35]</td>
</tr>
<tr>
<td>Energy Efficiency</td>
<td>WiseMAC &gt; Crankshaft &gt; SCP-MAC &gt; X-MAC &gt; B-MAC [47, 61, 42, 69], RI-MAC &gt; X-MAC [61], O-MAC &gt; B-MAC [21], BoX-MAC &gt; X-MAC [54], SCP-MAC &gt; T-MAC &gt; S-MAC [47]</td>
</tr>
<tr>
<td>Latency</td>
<td>WiseMAC &lt; X-MAC &lt; SCP-MAC &lt; B-MAC &lt; Crankshaft [61, 42, 69, 35], RI-MAC &lt; X-MAC [61]</td>
</tr>
</tbody>
</table>

Table 3.1: Known MAC protocol performance comparisons

into four classes in terms of coordination centricity and synchrony, and select the best exemplar from each one of these classes. From the relatively large space of protocol parameters, we restrict our attention to two parameters that potentially impact performance significantly and are exposed by virtually every duty-cycled MAC, namely wakeup interval and maximal queue size. We first present a novel framework for modeling the performance of these MAC protocols in a manner that accounts for inter-flow interference. Then, for a rich set of network density and traffic rate configurations, we exhaustively search the set of available protocol configurations for each of the representative protocols to determine the Pareto-optimal throughput, energy efficiency, and latency. This yields the basis to compare protocols in different parts of the configuration space, to design parameter adaptation methods, and to study tradeoffs.

### 3.1.1 Summary of the Results

The main findings of our analysis are summarized as follows:
• MAC protocols differ significantly in their ability to use the channel bandwidth. In particular, the sender-centric asynchronous protocol (BoX-MAC) achieves the highest throughput capacity across various network densities, while receiver-centric asynchronous protocol (RI-MAC) achieves the lowest throughput among the four.

• When network traffic increases while not exceeding the achievable capacity for a given network density, the energy efficiency either increases (as in BoX-MAC and RI-MAC) or remains almost constant (as in O-MAC and RI-MAC); when the traffic exceeds the limit, energy efficiency reduces accordingly. This implies that upper layers should adapt to traffic growth by increasing duty cycle rather than changing routes or frequency to improve efficiency as long as the traffic does not exceed the achievable capacity at that density.

• Our analysis corroborates the known tradeoff between energy efficiency and latency. We find additionally that (i) the tradeoff is relatively insignificant for high traffic, and (ii) receiver-centric synchronous protocols can achieve latency comparable to asynchronous protocols with higher energy efficiency. In other words, WSN applications are capable of adapting to various latency requirements efficiently using receiver-centric synchronous MACs.

• The combination of local synchrony and receiver centricity not only achieves the highest efficiency in the clique network, it also reduces the likelihood of collision among senders that are hidden terminals to each other in a general network.
Surprisingly, of the protocols evaluated one, O-MAC, shows a remarkably consistent behavior across much of the configuration space, always achieving the best or close to the best performance for all metrics that we have considered.

- The analytical results offer a basis for choosing MAC parameters. With O-MAC, for example, by configuring its frame length appropriately, O-MAC can reduce the queue length by 5 times at a cost of 10% in relative efficiency. Our analysis moreover shows that a local frame adaptation scheme yields close-to-optimal performance.

Last but not least, we validate the models by running a comprehensive set of experiments in TinyOS-2.x on TelosB sensor nodes in an office testbed and a public testbed. Our experimentation takes into consideration networks with and without hidden terminals, to study their impact on MAC performance. The experimental results match the analytical results nicely, with an error less than 20% and a correlation factor of 0.9 across all tested network configurations.

### 3.1.2 Organization of this Chapter

Related work on comparative performance of MACs is reviewed in Section 3.2. Section 3.3 presents the modeling framework as well as the representative MAC model for each class. Section 3.4 analytically evaluates the performance of these MAC protocols across the configuration space. Section 3.5 experimentally corroborates the analytical results. Moreover, a duty cycle adaptation scheme is proposed in Section 3.6. Extensions to the current framework are discussed in Section 3.7. Section 3.8 makes concluding remarks.

\(^2\)A major portion of this deviation is attributed to the abnormal behavior at high traffic in the implementation of RI-MAC. For BoX-MAC and O-MAC alone, the error is within 10%.
3.2 Related Work

The comparative study of MAC protocols in Chapter 2 only compares the achievable throughput given a duty cycle, thus the MAC models do not capture various performance metrics under different network and protocol configurations. The extended MAC models proposed in this chapter characterize MAC classes more comprehensively and practically across the configuration space.

Over the past years, there have been a few attempts at evaluating various MAC protocols under different network and protocol configurations. Boano et al [17] experimentally studied the throughput and energy consumption of several MACs under external interference for low data rate collection applications. However, their experiments only involved few sensor nodes, thus how their results apply to large networks is unknown. Hurni et al [38] studied how energy-efficient MACs adapted to fluctuating traffic in a linear network of eight nodes. The authors searched for the protocol that performed best under changing traffic using a fixed set of protocol configurations. In other words, they did not attempt to select, for each protocol being compared, protocol configurations that would respectively optimize its performance. Merlin et al [53] proposed two approaches to control the duty cycle so that the target rate of transmitted packets is reached, while the consumed energy is minimized. The first approach, called asymmetric additive duty cycle control (AADCC), employs a linear increase/linear decrease in the $ti$ value based on the number of successfully received packets. This approach is easy to implement, but it cannot provide an ideal solution. The second approach, called dynamic duty cycle control (DDCC) utilizes control
theory to strike a near-optimal balance between energy consumption and packet delivery successes. This approach involves non-negligible overhead in computing energy consumption and adapting to the optimal value.

Most related to our work is the study from Langendoen et al [47] in which the authors modeled and compared the performance of various MAC protocols at their Pareto-optimal points. However, their analysis has a fundamental limitation: it only applies to very low traffic case since they assume no interference would exist in a WSN application. In contrast to previous studies on MAC modeling, the present work explicitly take inter-flow interference into consideration, enabling us to more realistically characterize how different MAC protocols perform under diverse network settings.

3.3 Modeling MAC Protocols

In this section, we model the representative MAC of each class listed in Table 2.1. Model inputs are the network state, comprising traffic rates and network densities, and the MAC parameters that are critical for duty cycling nodes. Then, we propose a generic framework that enables us to model the most crucial aspects of MAC designs. Based on this framework, we instantiate detailed models of representative MAC in each category. Instantiating the framework for other MAC protocols in the same category has proven to be straightforward, thus we omit models of other MACs and focus on the exemplar MACs.

3.3.1 Network and Traffic Model

Since scheduling in MACs is fundamentally about single-hop communication, we consider the model where \( n \) nodes are randomly distributed in a clique network and
these nodes are grouped into $n/2$ pairs for unicast communication. Data traffic is generated independently by each source node at a rate of $r_{pkt}$ packets per second. That is, network traffic consists of $n/2$ single-hop flows with each flow sending data at a rate of $r_{pkt}$. In a clique network, the transmission of a node interferes with any other node, thus interference becomes a non-negligible issue in the network as traffic increases. The performance of each MAC protocol will be evaluated across a wide range of network densities ($n$) and traffic rates ($r_{pkt}$). Note that hidden terminals are not considered in the theoretical model, but their influence on MAC performance is evaluated via experiments in Section 3.5.3. Extensions to the current modeling for non-clique networks as well as for more sophisticated traffic patterns will be discussed in Section 3.7.

3.3.2 MAC Parameters

Towards comparing MAC protocols across a wide range of network settings, we restrict our attention on two protocol parameters, namely frame length and maximum queue size, that are exposed by virtually every duty-cycled MAC regardless of its coordination centricity and synchrony.

Frame length, denoted by $T_f$, defines the average time interval between consecutive wakeup times of a node. Reciprocally, $1/T_f$ is the wakeup frequency of a node. Maximum queue size, denoted by $Q_{max}$, represents the maximum number of outgoing packets that can be buffered at a node, which determines the maximum length of back-to-back packets that may be transmitted during each wakeup. Thus, these two parameters are critical in determining radio active time, corresponding to the major
energy consumption of a node; however, the impact of these two parameters on MAC performance at various network configurations is yet to be studied.

Of course, in addition to frame length and maximum queue size, a number of other parameters are tunable in a MAC protocol, such as synchronization interval, contention window size, backoff window size, maximum number of retransmissions and etc. Optimization of those parameters have been studied extensively either for individual MACs [57, 69, 54, 61] or for MAC comparison [47]. Thus, this work focuses on the less investigated but important MAC parameters \(T_f\) and \(Q_{\text{max}}\), while employing default values for other MAC parameters (as listed in Table 3.3) that would offer the optimal performance.

### 3.3.3 The Framework

Although duty-cycled CSMA-based MAC protocols differ in their scheduling scheme and the associated parameters, they share two features: the periodicity of radio wakeup and CSMA contention scheme. The framework characterizes these protocols by analyzing the radio events that may occur within each frame for the sender and receiver, separately. In computing the event probabilities and durations, MACs take on their own values in a set of MAC-specific parameters in keeping with the difference in rendezvous schemes. Table 3.2 summarizes these MAC-specific parameters used in the framework. As we will see in Section 3.3.4, to instantiate the framework to model a particular MAC, one only needs to determine the value of MAC-specific parameters since the computation of event probabilities remain the same across protocols.

First, a sender may perform one of two events per frame: (i) it attempts to transmit but fails to win the CSMA contention \((p_{fl}, T_{fl})\); and (ii) it transmits after
successful contention \((p_{tx}, T_{tx})\). Associated with each event is its probability to occur in a frame and the duration of radio time. For instance, a transmission event occurs for a duration of \(T_{tx}\) with a probability of \(p_{tx}\) in each frame. Note that the transmission may not be received successfully if more than one sender claims success in the contention and transmits.

Correspondingly, a receiver may perform one of four events per frame: (i) it polls the channel but detects no transmission \((p_{idl}, T_{idl})\); (ii) it overhears a transmission that is not destined to itself \((p_{ovr}, T_{ovr})\); (iii) it receives a packet successfully \((p_{suc}, T_{rx})\); and (iv) it receives a corrupted packet \((p_{col}, T_{rx})\). Note that the radio time spent in receiving corrupted and non-corrupted transmissions are approximately the same, thus both durations are denoted by \(T_{rx}\). Since nodes in the network are paired for communication, the following equation

\[
p_{tx} = p_{suc} + p_{col}
\]  

(3.1)

holds for all sender-receiver pairs.

Next, we detail the behavior of the sender and receiver within a frame to derive these probabilities.

**Sender.** As data is generated at a rate of \(r_{pkt}\) packets per second, assuming senders only access the channel upon the arrival of new data, the probability for a sender to contend for the channel in a frame is denoted by \(p_t\), where \(p_t \in [0, 1]\) and

\[
p_t = \min(r_{pkt} \cdot T_f, 1).
\]  

(3.2)

When more than one packets have to be sent within a frame (i.e., \(r_{pkt} \cdot T_f > 1\)), the probability to contend for the channel remains as 1. Extra packets are queued in the buffer and transmitted in a back-to-back manner with a minimal gap in between,
which avoids contending the channel for each packet. Accordingly, we compute the event probabilities for a sender as follows:

\[ p_{tx} = p_t \cdot p_a \quad \text{and} \quad p_{ft} = p_t \cdot (1 - p_a), \]  
(3.3)

where \( p_a \) is the probability that a sender wins the contention.

We derive \( p_a \) based on the typical CSMA procedure adopted by most WSN MAC protocols. The sender first randomly selects one out of \( w \) timeslices in the contention window and then monitors the channel till the chosen timeslice to ensure that no other transmission occurs. The length of each timeslice is denoted by \( t_{cw} \). If any transmission is detected before the chosen timeslice, the sender aborts its transmission attempt; otherwise, it immediately starts the preamble or data transmission in its chosen timeslice. We define \( \hat{\epsilon} \) as the expected number of active contenders with respect to a sender. Parameters \( w \) and \( \hat{\epsilon} \) are MAC specific, the values of which will be discussed in each individual MAC.

Let the probability that a sender successfully delivers a packet to the receiver be \( p_s \). Note \( p_s \leq p_a \) in half-duplex mode, since nodes may happen to pick the same timeslice and transmit simultaneously without detecting each other, resulting in a collision at receivers. Given MAC-specific \( \hat{\epsilon} \) and \( w \), probabilities \( p_a \) and \( p_s \) are expressed as:

\[ p_a = \frac{1}{w} \sum_{j=1}^{w} \left(1 - \frac{j-1}{w}\right)^{\hat{\epsilon}} \quad \text{and} \quad p_s = \frac{1}{w} \sum_{j=1}^{w} \left(1 - \frac{j}{w}\right)^{\hat{\epsilon}}, \]  
(3.4)

respectively. Note that the difference between \( p_a \) and \( p_s \) is that in the event corresponding to \( p_s \) no contending node chooses a timeslice (prior to or) equal to \( j \).

**Receiver.** For the receiver, we can derive corresponding event probabilities in a similar way. We define \( \hat{\eta} \) to be the expected number of potential senders when a receiver wakes up; it is also MAC dependent. Thus, in the event where no transmission
Table 3.2: MAC-specific parameters in the framework

is attempted by any potential sender upon a receiver wakeup, \( p_{idt} \) equals \((1 - p_t)^\hat{\eta}\). The rest of Eq. (3.5) is self-explanatory.

\[
\begin{align*}
    p_{idt} &= (1 - p_t)^\hat{\eta}, & p_{ovr} &= 1 - (1 - p_t)^\hat{\eta} - p_t \cdot p_a, \\
    p_{suc} &= p_t \cdot p_s, & p_{col} &= p_t \cdot (p_a - p_s).
\end{align*}
\] (3.5)

If a packet is not delivered successfully, it will be buffered in sender’s queue for retransmission. Recall that \( Q_{max} \) is the maximum queue size at each node. Let \( Q_{pkt} \) be the expected number of packets in the queue. On average, a sender transmits \( Q_{pkt} \) packets back-to-back once it wins the channel contention, which is modeled by the product of the number of accumulated packets over a frame, \( r_{pkt} T_f \), and the expected number of frames required to successfully deliver the packet, \( 1/p_{suc} \):

\[
Q_{pkt} = \min(\frac{r_{pkt} \cdot T_f}{p_{suc}}, Q_{max}) = \min(1/p_{suc}, Q_{max}).
\] (3.6)

**Performance Metrics.** Having analyzed the event probabilities at the sender and receiver, we now proceed to model the performance metrics of interest, including the sender and receiver duty cycle (\( \psi_s \) and \( \psi_r \), respectively), delivery ratio (\( \tau \)), one-hop latency (\( \delta \)), and energy efficiency (\( e \)) for each pair of nodes.
First, the duty cycle of the sender is modeled as

\[ \psi_s = \frac{1}{T_f} (p_{tx} \cdot T_{tx} + p_{fl} \cdot T_{fl}), \quad (3.7) \]

and the duty cycle of the receiver is

\[ \psi_r = \frac{1}{T_f} (p_{idl} \cdot T_{idl} + p_{ovr} \cdot T_{ovr} + p_{tx} \cdot T_{rx}). \quad (3.8) \]

Constraints on duty cycles of the sender and receiver are \( 0 < \psi_s, \psi_r \leq 1 \). For ease of presentation, we use the metric of average duty cycle per flow, \( \psi \), where

\[ \psi = \frac{\psi_s + \psi_r}{2}. \quad (3.9) \]

Second, the per flow delivery ratio, \( \tau \), equals the ratio of the number of successfully delivered packets to the total number of generated packets:

\[ \tau = \frac{p_{suc} \cdot Q_{pkt}}{T_f \cdot r_{pkt}}. \quad (3.10) \]

Accordingly, the throughput per flow is the product of \( \tau \) and source rate \( r_{pkt} \).

Third, latency is defined as the expected time interval from the first attempt to send a packet to successfully delivery of the packet. Upon a transmission attempt, the expected time before the next rendezvous is \( T_f/2 \), thus the average latency for a packet is

\[ \delta = \left( \frac{1}{\tau} - \frac{1}{2} \right) \cdot T_f. \quad (3.11) \]

Lastly, we compute energy efficiency in units of \( dB \) as the ratio of energy expended in useful communication to the total energy expended. Thus, \( 0 dB \) equals 100% efficiency. \( E_{pkt} \) and \( E_{radio} \) stand for the energy consumption per packet and the energy consumption rate of the active radio, respectively. \( t \) refers to the length of
Table 3.3: Notation and typical values of constants

time interval for which we compute the efficiency. It is worth noticing that $e$ offers an explicit evaluation on the cost effectiveness of a MAC protocol in terms of throughput and energy.

$$e = 10 \log \left( \frac{\tau \cdot r_{pkt} \cdot t \cdot E_{pkt}}{(\psi_s + \psi_r)/2 \cdot t \cdot E_{radio}} \right).$$

(3.12)

In summary, our framework consists of Eq. (3.1)-(3.12). We are now ready to analyze the performance of each MAC protocol by instantiating values of the MAC-specific parameters summarized in Table 3.2.

### 3.3.4 Case Study of MAC Models

In this section, we elaborate on how to instantiate parameters in Table 3.2 for four exemplar MACs. Table 3.3 summarizes typical values of MAC constants used in the framework.
O-MAC

O-MAC [21] is receiver-centric and locally synchronous: nodes use seeds to generate pseudo-random wakeup schedules to stagger their wakeup times. Receivers broadcast their seeds periodically so that neighboring nodes can compute their corresponding wakeup times locally. Via the local synchrony, senders with pending data need only to wake up just before their intended receiver does.

Regarding the operation of back-to-back transmission, i.e., queued packets are sent continuously upon the successful delivery of the first packet, we introduce the notion of slot. The slot starts with a contention window of length $T_{cw}$ followed by packet transmissions. The number of timeslices, $w$, is derived by

$$w = \frac{T_{cw}}{t_{cw}}. \quad (3.13)$$

Since queued packets are sent consecutively with minimum gap ($T_{gap}$), we consider the slot length to be

$$T_{slot} = T_{cw} + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil, \quad (3.14)$$

where $T_{pkt}$ is the time to transmit a single packet. Recall that the number of queued packets, $Q_{pkt}$, is determined by Eq. (3.6). One constraint on the parameter is that a complete packet sequence must fit into one frame length, i.e.

$$T_{radio} + T_{slot} \leq T_f, \quad (3.15)$$

where $T_{radio}$ is the time duration for switching the radio on and off.

For ease of modeling, we assume that slots are aligned among nodes. Given that receivers independently wake up at random slots, let $I(i)$ be the indicator function of
node $i$ contending for the channel in a given slot, where

$$I(i) = \begin{cases} 1, & i \text{ attempts to transmit in a given slot;} \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the expected number of contenders with respect to a sender in any slot, $\hat{\epsilon}$, is computed by Eq. (3.16).

$$\hat{\epsilon} = E\left[\sum_{i=1}^{n/2-1} I(i)\right] = \sum_{i=1}^{n/2-1} E[I(i)] = \left(\frac{n}{2} - 1\right) \cdot \frac{T_{\text{slot}}}{T_f} \cdot p_t,$$

where $(T_{\text{slot}}/T_f) \cdot p_t$ is the probability that a sender contends for the channel in a given slot since the probability for a receiver to wake up in a given slot is $T_{\text{slot}}/T_f$.

Similarly, the expected number of potential senders with respect to a receiver is

$$\hat{\eta} = \frac{n}{2} \cdot \frac{T_{\text{slot}}}{T_f}.$$  (3.17)

Although O-MAC requires exchanging time information at least once every $T_{\text{sync}}$ interval in order to keep the communicating pairs synchronized, $T_{\text{sync}}$ can be up to several minutes for the needed accuracy in local synchrony [21]. Moreover, a node’s seed information can be embedded with the synchronization message. The guard time to tolerate synchronization error can be expressed by [47]

$$T_{\text{guard}} = 4T_{\text{sync}} \cdot \theta,$$  (3.18)

where $\theta$ denotes the frequency tolerance (ppm) of clock. When packet arrival rate is lower than $1/T_{\text{sync}}$ a separate synchronization message exchange is demanded, which results in an extra duty cycle of

$$\psi_{\text{sync}} = \frac{T_{\text{radio}} + T_{\text{guard}} + T_{\text{cw}} + T_{\text{pkt}}}{T_{\text{sync}}}.$$  (3.19)

Otherwise, synchronization messages are exchanged by attaching several bits to data packets.
To summarize, each event duration at sender side is computed as:

\[
T_{tx} = T_{radio} + T_{cw}/2 + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil ,
\]

\[
T_{fl} = T_{radio} + T_{cw}/2 .
\]  \hspace{1cm} (3.20)

For a receiver node, the duration of each event is computed in Eq. (3.21). Here, \( T_{hdr} \) denotes the processing time to read the header of a packet. For packet-based radios, \( T_{hdr} \approx T_{pkt} \).

\[
T_{idl} = T_{radio} + T_{guard} + T_{cw} ,
\]

\[
T_{ovr} = T_{radio} + T_{guard} + T_{cw}/2 + T_{hdr} ,
\]  \hspace{1cm} (3.21)

\[
T_{rx} = T_{radio} + T_{guard} + T_{cw}/2 + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil .
\]

**BoX-MAC**

BoX-MAC\(^3\) [54] is a sender-centric asynchronous protocol, where nodes independently and periodically poll the channel for activity. Each sender wakes up its receiver by sending a preamble that can be as long as \( T_f \).

In BoX-MAC, the expected preamble duration for a sender is \( T_f/2 \), during which on average half the senders with new data will contend for the channel. However, only the earliest contender will detect a clear channel and proceed to preamble transmission that will occupy the channel until the destined receiver becomes awake. Thus, we define this window of length \( T_f/2 \) as the *virtual contention window*, which is different from \( T_{cw} \). The latter is sender carrier sensing time, i.e., the upper bound of time an individual sender monitors channel before it transmits, which is usually much shorter.

\(^3\)There are two types of BoX-MAC: BoX-MAC-1 and BoX-MAC-2. In this work, we evaluated BoX-MAC-2 because BoX-MAC-1 is favorable for low data-rate applications.
than $T_f/2$. Hence, the number of timeslices for the virtual contention window is

$$w = \left\lfloor \frac{T_f}{2t_{cw}} \right\rfloor. \quad (3.22)$$

The probability that a sender contends for the channel within a virtual contention window is $1/2 \cdot p_t$. We can compute $\hat{\epsilon}$ in the same way as Eq. (3.16), where

$$\hat{\epsilon} = E\left[\sum_{i=1}^{n/2-1} I(i)\right] = \sum_{i=1}^{n/2-1} E[I(i)] = \left(\frac{n}{2} - 1\right) \cdot \frac{1}{2} \cdot p_t. \quad (3.23)$$

The expected number of potential senders is

$$\hat{\eta} = \frac{1}{2} \cdot \frac{n}{2}. \quad (3.24)$$

As for BoX-MAC, if a node fails to transmit with probability $p_{fl}$ (in Eq. (3.3)), each time it has to backoff (without switching off the radio) for a duration of $T_{bf}$.

$$T_{tx} = T_{radio} + T_f/2 + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil,$$
$$T_{fl} = T_{radio} + T_{cw}/2 + T_{bf}. \quad (3.25)$$

The event durations of a receiver are calculated as follows, where $T_{cs}$ denotes the LPL-based channel polling time.

$$T_{idl} = T_{radio} + T_{cs},$$
$$T_{ovr} = T_{radio} + T_{cs}/2 + T_{pkt},$$
$$T_{rx} = T_{radio} + T_{cs}/2 + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil. \quad (3.26)$$

One constraint is that the length of frame length cannot be less than the time consumed for data transmission.

$$T_{radio} + T_f/2 + (T_{pkt} + T_{gap}) \cdot \lceil Q_{pkt} \rceil \leq T_f. \quad (3.27)$$
SCP-MAC

SCP-MAC [69] adopts LPL channel polling scheme, however, unlike LPL it synchronizes the polling times of neighboring nodes. Transmitting a packet in SCP-MAC involves two steps: first, the sender transmits a short wakeup tone \((T_{\text{tone}})\) timed to intersect with the receiver’s channel polling; second, after waking up the receiver the sender transmits the actual data packet. To reduce the probability of collision, a sender has to perform two-phase contention. Before sending the tone, a node contends for the channel by randomly choosing a timeslice within the first contention window \((T_{cw1})\). An idle channel allows the node to proceed: entering the second contention phase \((T_{cw2})\) and sending the wakeup tone. If the node detects a clear channel in the second phase it starts sending wakeup tone and data; otherwise (due to another node transmitting a tone first), it aborts transmission until the next frame and waits to receive potential traffic.

As a synchronous protocol, SCP-MAC requires that at least one message is sent every \(T_{\text{sync}}\) period in order to keep the neighboring nodes synchronized, which results in a guard time of \(T_{\text{tone}}\) calculated in [69], where

\[
T_{\text{tone}} = 4T_{\text{sync}} \cdot \theta/(n/2 - 1) + T_{\text{mtone}}. \tag{3.28}
\]

\(T_{\text{mtone}}\) is the minimum tone interval specified in the protocol. When arrival packet interval is longer than \(T_{\text{sync}}\), an explicit synchronization message exchange is required once every \(T_{\text{sync}}\) time period, which results in an extra duty cycle of \(\psi_{\text{sync}}\), where

\[
\psi_{\text{sync}} = \frac{T_{\text{radio}} + T_{cw1} + T_{\text{tone}} + T_{cw2}/2 + T_{\text{pkt}}}{T_{\text{sync}}}. \tag{3.29}
\]
To account for the two-phase CSMA contention, let $A$ be the event that a sender successfully secures the channel at the end of the second carrier sensing phase. According to the law of total probability, we divide the sample space into three events, namely $B_1$, $B_2$, and $B_3$. Event $B_1$ denotes that the sender has exclusively secured the channel in the first contention; thus there is no other competitors during the next contention phase. In event $B_2$, the sender has detected a clear channel simultaneously with other contenders during the first phase, thus, it will participate in the next phase. Event $B_3$ represents that the sender fails to secure the channel in the first contention. Thus, the overall probability $p_a$ is computed in the following equations:

$$p_a = P_r(A) = \sum_{i=1}^{3} P_r(A|B_i)P_r(B_i),$$

$$= p_{s,1} \cdot 1 + p_{a,2} \cdot (p_{a,1} - p_{s,1}) + 0 \cdot (1 - p_{a,1}), \quad (3.30)$$

where $p_{s,i}$ and $p_{a,i}$ respectively stand for the probability that a node exclusively and non-exclusively secures the channel in the $i$th contention phase. They are expressed by

$$p_{a,i} = \frac{1}{w_i} \sum_{j=1}^{w_i} (1 - \hat{\epsilon}_i)^j, \quad (3.31)$$

$$p_{s,i} = \frac{1}{w_i} \sum_{j=1}^{w_i} (1 - \hat{\epsilon}_i)^j,$$

$$w_i = T_{cw}/t_{cw},$$

$$\hat{\epsilon}_1 = (n/2 - 1) \cdot p_t,$$

$$\hat{\epsilon}_2 = \hat{\epsilon}_1 \cdot (p_{a,1} - p_{s,1}). \quad (3.32)$$

Likewise, the overall probability $p_s$ can be derived as:

$$p_s = p_{s,1} + p_{s,2} \cdot (p_{a,1} - p_{s,1}). \quad (3.33)$$
Due to the fact that all neighboring nodes wake up simultaneously, the expected number of potential senders with respect to a receiver is $\hat{\eta} = n/2$. Event durations of the sender and receiver are as follows.

\[
T_{tx} = T_{radio} + T_{cw1} + T_{tone} + T_{cw2}/2 + (T_{pkt} + T_{gap}) \cdot [Q_{pkt}],
\]

\[
T_{fl} = T_{radio} + T_{cw1} + (T_{tone} + T_{cw2})/2,
\]

\[
T_{idl} = T_{radio} + T_{tone}/2 + T_{cw2},
\]

\[
T_{ovr} = T_{radio} + T_{tone}/2 + T_{cw2}/2 + T_{hdr}
\]  \hspace{1cm} (3.34)

\[
T_{rx} = T_{radio} + T_{tone}/2 + T_{cw2}/2 + (T_{pkt} + T_{gap}) \cdot [Q_{pkt}].
\]

The constraint on the frame length is

\[
T_{radio} + T_{cw1} + T_{tone} + T_{cw2} + (T_{pkt} + T_{gap}) \cdot [Q_{pkt}] \leq T_f.
\]

**RI-MAC**

RI-MAC [61] is the representative for receiver-centric asynchronous protocols, where communication rendezvous is initiated by the receiver. Unlike O-MAC, receivers in RI-MAC periodically broadcast beacons to announce their wakeup moments and invite incoming packets in lieu of relying on time synchronization. Once data arrives, the sender has to keep its radio active in receive mode waiting for beacon from the intended receiver and contend for the channel upon receiving each beacon.

Let $T_{bn}$ be the length of time consumed in sending out a beacon. MAC-specific parameters of RI-MAC can be computed in a similar way of O-MAC since they both
belong to the receiver-centric coordination scheme.

\[ w = T_{cw}/t_{cw}, \]

\[ T_{slot} = (T_{bn} + T_{cw} + T_{pkt}) \cdot [Q_{pkt}], \]

\[ \hat{e} = \left( \frac{n}{2} - 1 \right) \cdot \frac{T_{slot}}{T_f} \cdot p_t, \]

\[ \hat{\eta} = \frac{n}{2} \cdot \frac{T_{slot}}{T_f}. \] (3.35)

The event durations of the sender and receiver are close to formulae of BoX-MAC given that energy cost in transmit and receive mode are comparable. Hence, the rest of MAC-specific parameters of RI-MAC are as follows.

\[ T_{tx} = T_{radio} + T_f/2 + (T_{bn} + T_{cw}/2 + T_{pkt}) \cdot [Q_{pkt}], \]

\[ T_{fl} = T_{radio} + T_f/2 + T_{bn} + T_{cw}/2, \]

\[ T_{idl} = T_{radio} + T_{bn} + T_{cw}, \]

\[ T_{ovr} = T_{radio} + T_{bn} + T_{cw}/2 + T_{hdr}, \]

\[ T_{rx} = T_{radio} + (T_{bn} + T_{cw}/2 + T_{pkt}) \cdot [Q_{pkt}]. \] (3.36)

The constraint on the frame length is

\[ T_{radio} + T_f/2 + T_{slot} \leq T_f. \] (3.37)

Recall that notation and values of MAC constants used in the framework can be found in Table 3.3.

### 3.4 Numerical Results of MAC Performance

Based on the instantiated framework, the performance of exemplar MAC protocols is now evaluated across the network configuration space. First, we compare
throughput capacity and energy efficiency of MAC protocols. Second, we analyze the tradeoff between energy efficiency and latency. Third, we study the queue size along with its tradeoff with throughput and energy efficiency. Lastly, the sensitivity analysis on MAC frame length motivates a local adaptation scheme for optimizing MAC performance.

In terms of performance evaluation, we compare the four classes at their respectively optimal points by exhaustively searching the available set of the parameters for each representative MAC at various network configuration. Each MAC model is evaluated across a broad range of traffic rates from extremely low to very high traffic rates, which are 0.001, 0.01, 0.05, 0.1, 0.5, 1, 4, 8, 12, 16, and 20 packets per second, and network densities that range from 1 to 10 flows. The maximum queue size of each protocol is set to 10. For each MAC at each network configuration, we select the minimal duty cycle among results that has the best throughput by exhaustively searching over a range of frame lengths between 10 and 5000 milliseconds with a step size of 10 milliseconds.

3.4.1 Energy Efficiency and Throughput Capacity

Reliable packet delivery is one of the most critical requirements for a MAC protocol. As can be expected, the throughput capacity of a duty-cycled MAC protocol depends on the quality as well as efficiency of its scheduling mechanism. To characterize the ability of four classes to use channel bandwidth, we compare the delivery ratio of MAC protocols at diverse network configurations.

Fig. 3.1 (a) shows the comparison of delivery ratio.\textsuperscript{4} We observe that the delivery ratio of a MAC protocol remains at 100% until it reaches its own capacity with respect to

\textsuperscript{4}Figures are best viewed in color.
to the number of flows. Specifically, BoX-MAC provides the best throughput capacity, followed by SCP-MAC, O-MAC and RI-MAC. When the traffic is high, BoX-MAC operates almost like full duty-cycle CSMA with little rendezvous overhead since the length of preamble reduces significantly as nodes wake up more often. Although BoX-MAC and RI-MAC are both asynchronous protocols, the high traffic condition prevents RI-MAC from effectively broadcasting beacons to invite each transmission due to the high channel occupancy. Thus, RI-MAC suffers from the use of receiver beacons at high traffic and degrades most sharply as traffic increases.

The throughput of SCP-MAC and O-MAC are between that of BoX-MAC and RI-MAC due to the overhead introduced by their slotted communication. However, they both achieve much higher delivery ratio than RI-MAC since the overhead of slotted communication is not as involved as the beacons. Senders in SCP-MAC attempt to access the channel during the same period of time, but the two-phase contention scheme gracefully resolves collision even at high traffic rates. O-MAC is a close third: its slot-based scheduling scheme only becomes a bottleneck at high traffic. But note that the 10% discrepancy in delivery ratio between O-MAC and SCP-MAC will disappear if O-MAC is enhanced with the two-phase contention scheme.

In addition to throughput, energy efficiency is another critical requirement for MAC protocols, thus a comprehensive understanding of the relation between the two is essential. Fig. 3.1 (b) depicts the minimum duty cycles consumed at four MAC protocols to achieve their respectively optimal throughput. Asynchronous MACs show higher energy consumption than synchronous MACs due to their costly rendezvous mechanism. Synchronous MACs minimize energy wasted in preamble transmission
or idle listening by the use of piggybacked synchronization. Thus, given the delivery ratio and minimum duty cycle in Fig. 3.1, we further plot the achievable energy efficiency in units of $dB$ across the same network configurations in Fig. 3.2 (a).

As we see in Fig. 3.2 (a), receiver-centricity combined with synchrony achieves the best energy efficiency, i.e., across almost all evaluated configurations, O-MAC is the most energy efficient. The observation is in line with the conclusion made for low data rate applications [47], although in that work it was exemplified with WiseMAC, which belongs to the same class as O-MAC. The efficiency of SCP-MAC becomes a second since its synchronized wakeup cycles introduce more idle listening during channel contention than O-MAC. Nevertheless, both O-MAC and SCP-MAC maintain flatter energy efficiency prior to reaching their capacities. Asynchronous MACs suffer from their costly rendezvous mechanisms, but improve in efficiency as traffic increases: the overhead of rendezvous decreases with the traffic rate.

Lastly, in terms of relationship between energy efficiency and throughput capacity, before reaching the respective capacity limit, as throughput increases the energy
efficiency of a MAC protocol either increases (as BoX-MAC and RI-MAC) or remains constantly (as O-MAC and SCP-MAC); but when the aggregate traffic, i.e., the product of the number of flows and flow rate, approaches the MAC capacity, both throughput and energy efficiency no longer increase or even degrade significantly. Note that the increase of energy efficiency of O-MAC and SCP-MAC at very low traffic rates are caused by the upper bound of frame length (5000 milliseconds) since synchronous MACs favor utilizing longer frame lengths to improve efficiency.

The relationship implies that energy efficiency can be improved by adapting frame length to traffic changes instead of alternative approaches, such as changing routes or switching frequencies, as long as the local traffic has not reached the capacity; however, when the traffic increases above the MAC-specific throughput capacity, the alternative approaches should be applied.
3.4.2 Tradeoff with Energy Efficiency: Latency

Although synchronous MACs consume less energy, it has been argued previously that their energy efficiency comes at the cost of larger latency [42, 35]. This essentially follows from the different preferences for choosing frame lengths between asynchronous and synchronous protocols. Nevertheless, in this section we show that this argument is insufficient in that synchronous protocols can be both superior in energy efficiency and competitive in latency across a broad range of traffics. In particular, we show that O-MAC and SCP-MAC can trade their energy efficiency for latency, achieving a latency comparable to that of asynchronous protocols while still outperforming them in efficiency.

In Fig. 3.2 (b), each data point is obtained as follows: for a given traffic rate and network density, in addition to optimizing energy efficiency subject to maximizing throughput we introduce another constraint, denoted by $T^*$, that demands the per-hop latency is no greater than the reciprocal of incoming packet rate. When traffic rates are 16 and 20 packets per second, the MAC capacities have been exceeded, thus we relax the latency constraint to be $1/12$ second. As we can see, because O-MAC and SCP-MAC have to operate at a shorter-than-optimal frame length to match constraint $T^*$ their energy efficiency reduces substantially as compared to their most energy efficient configurations, especially at low traffic. However, even then the efficiency of O-MAC is still at least $2.5 \times (\approx 4 \text{ dB})$ better than BoX-MAC and RI-MAC. Additionally, the tension between energy efficiency and latency of O-MAC reduces as traffic increases since the tradeoff becomes trivial at high traffic.
3.4.3 Tradeoff with Energy Efficiency: Queue Size

RAM is a key resource constraint in many WSN node platforms; for example, the TelosB [6] mote contains only 10KB of RAM. Packet queues comprise the bulk of MAC protocol memory usage. We now study how the expected queue size is influenced by configuring MAC protocols to operate at different frame lengths. We plot the frame tuning results for O-MAC, the most energy-efficient protocol among the four, in Fig. 3.3, where the queue size and energy efficiency are both shown on the Y axes. The traffic considered has 3 flows with each one sending 8 packets per second, which is within the throughput capacity of O-MAC. The frame length increases from 10 to 5000 milliseconds by a unit of 50 milliseconds each time. We observe that both energy efficiency and queue size grow as the frame length increases until the maximum queue size is reached, in which case $Q_{max}$ is 20. The vertical line drawn in the figure points out the queue size and frame length where the highest gradient of energy efficiency curve is reached, suggesting that a minor reduction in energy efficiency (0.5 dB) results in a notable decrease in queue size.

Figure 3.3: Tradeoff between queue size and energy efficiency for O-MAC.
Figure 3.4: Performance of SCP-MAC under various queue sizes (a) delivery ratio and (b) energy efficiency.

Inversely, to understand the impact of maximum queue size on throughput and energy efficiency, we study the sensitivity of $Q_{max}$ at various network configurations for each MAC protocol. Fig. 3.4 and Fig. 3.5 plot the delivery ratio and energy efficiency of SCP-MAC and O-MAC when $Q_{max}$ are set to be 2, 4, and 20, respectively. We note that the performance of SCP-MAC is most sensitive to the reduced queue size among four protocols. As can be expected, in order to compensate for the high contention incurred by synchronized wakeup cycles, SCP-MAC favors larger queues to deliver packets back-to-back once the sender successfully secures the channel. In contrast, O-MAC is less affected by queue size: its energy efficiency reduces by less than 0.5 $dB$ between queue size of 4 and 20. Results for other two MACs are omitted here since their deductions in performance are less than or comparable with O-MAC across all configurations. *Therefore, the most energy-efficient of the protocols under study, O-MAC, is capable of reducing its queueing memory by up to a factor of 5 at a cost of losing at most 10% in relative efficiency.*
Figure 3.5: Performance of O-MAC under various queue sizes (a) delivery ratio and (b) energy efficiency.

3.5 Experimental Validation of MAC Models

We experimentally corroborated the model-based analytical results for three protocols across a wide range of network configurations. We performed two sets of experiments. In the first set, which was based on a clique network in our office testbed, we experimented with different traffic rates and densities to validate the soundness of the MAC models. To verify that our results are applicable to different environments, our second set involved experiments in a non-clique network on a public testbed, Kansei [10]. The testbed represents a more general network, in which senders can become hidden terminals to one another.

3.5.1 Experiment Configuration

Our experiments evaluated existing implementations of O-MAC and Box-MAC\textsuperscript{5} in TinyOS-2.1 and of RI-MAC in TinyOS-2.0.2. All experiments were performed on

\textsuperscript{5}The BoX-MAC-2 version we tested is included as the default MAC protocol for TinyOS-2.1
the TelosB platform. The original SCP-MAC implementation was for Mica2 motes in TinyOS-1.x. A recent version of SCP-MAC in MLA [42] has not yet implemented the overhearing avoidance feature described in the paper, which negatively impacts the efficiency of SCP-MAC. Thus we could not include SCP-MAC in the experiments.

To monitor the energy cost of a node, we implemented a software module in the TinyOS CC2420 driver that was used by all three MAC protocols to measure their cumulative duty cycle. The radio is considered active upon detecting the oscillator stabilization signal and inactive upon detecting the radio stop signal. The small power difference between the transmission mode (52.2 mW) and reception mode (56.4 mW) is ignored in computing energy expense; the average, 54.3 mW, is used as the power of an active radio. The useful energy cost is computed as the product of the energy cost per packet, denoted as $E_{pkt}$, and the total number of unique packets received, where $E_{pkt} = 40 \cdot 32 \cdot 54.3 \approx 69 \mu W$, as each packet is 40 bytes long, including the 802.15.4 header, and each byte takes approximately 32 µs to transmit. The total energy cost is computed by the product of the average radio power, the duty cycle, and the experiment duration.

We configured each MAC as follows: packets were buffered in a FIFO queue and retransmission was enabled for up to 5 times per packet. Sequence numbers were attached to data packets to filter out duplicate packets. Since traffic was one-way, sender frame length was set to a large constant, 100 seconds. Each experiment was conducted for a duration of 15 minutes. The measurement of the total energy cost began when all flows had started transmitting data.
3.5.2 Model Validation in Clique Networks

Considering the complexity of experimentation, for each MAC protocol four flow counts (from 1 to 4), five traffic rates (1, 4, 8, 16, and 32 packets per second), and six different frame lengths (25, 50, 100, 200, 500, and 1000 ms) were instantiated, leading to 120 experiments per MAC protocol. For a given flow count and flow rate, we selected the best delivery ratio and energy efficiency by searching over the six frame lengths. To better compare the results, we plot analytical and experimental results on the performance of three MAC protocols under the same network configurations in Fig. 3.6 and Fig. 3.7, respectively.

In terms of delivery ratio, the outermost contour line in Fig. 3.7 (a) shows that the delivery rate of RI-MAC degrades quickly as the aggregate traffic increases. The advantage of using of beacons in RI-MAC thus turns out to be limited to relatively low traffic. We noted a significant number of failures in sending out beacons in RI-MAC at...
Figure 3.7: Experimental performance of O-MAC, BoX-MAC, and RI-MAC at diverse network configurations (a) delivery ratio and (b) energy efficiency.

High traffic rates\textsuperscript{6}, which led to much lower throughput than expected. In contrast, both O-MAC and BoX-MAC achieve 100% delivery ratio in almost all test cases: with 4 flows and 32 packets per second, the delivery ratio of O-MAC and BoX-MAC reduce to 84% and 95%, respectively. Across the evaluated network configurations, the average error of delivery ratio for both O-MAC and BoX-MAC are 1%, while the average error for RI-MAC is 15%.

Fig. 3.7 (b) shows the best energy efficiency of each MAC protocol at given network configurations, which are in line with the trend in Fig. 3.6 (b). The correlation coefficient between experimental and analytical results are 0.95 for BoX-MAC and RI-MAC, and 0.85 for O-MAC. Although the major radio activities have been modeled by the framework, certain engineering overheads still exist in the protocol implementation, resulting in an energy efficiency that is approximately 2 dB lower than that in our analysis. Thus, the error in energy efficiency for three protocols is around

\textsuperscript{6}We contacted the authors of RI-MAC about this issue and confirmed that the current implementation of RI-MAC was not optimized or tested under high traffic rates.
20\%. For the purpose of establishing the robustness of our results to environment, we repeated the experiments of Fig. 3.7 on Kansei testbed that has about 400 TelosB motes deployed. The results are essentially identical.

## 3.5.3 Comparison in Non-Clique Networks

We conclude with experiments on Kansei testbed [10] to verify whether the comparative performance of the MACs shows the same trends in a general network, where nodes can be hidden terminals with respect to one another. In terms of experiment setup, we found it nontrivial to control hidden terminals in the network while still ensuring reliable channels between the respective senders and receivers. So we emulated the desired topology by forcing the radio to always return a clear channel after Clear Channel Assessment (CCA), which is the very essence of hidden terminals. Thus senders detect a free channel even when another sender is transmitting and create artificial collisions.

We adopted a similar experiment methodology as the ones in the previous section. The only difference is that, instead of increasing the number of flows from 1 to 4, we fixed the total number of flows to 4 but increased the number of flows with hidden-terminal senders from 1 to 4. For each network configuration that consisted of a given flow rate and number of senders whose CCA were turned off, we searched for the optimal energy efficiency by varying the frame length over six values (25, 50, 100, 200, 500, and 1000 ms). Fig. 3.8 shows the percentage change in optimal energy efficiency for each network configuration, which was computed against the result without hidden terminals at the corresponding flow rate under 4 flows in Fig. 3.7 (b).
O-MAC is least affected by hidden terminals among the three. The change in energy efficiency for O-MAC fluctuates between $-4\%$ and $4\%$ across all tested network configurations. RI-MAC is slightly affected by hidden terminals due to the use of beacons and its efficiency drops by $8\%$ when the aggregate traffic reaches the peak. BoX-MAC is most sensitive to the existence of hidden terminals; the drop in energy efficiency increases with the aggregate traffic and peaks at $18\%$.

At a high level, although there are small fluctuations in optimal energy efficiency of O-MAC and RI-MAC, the general trend is clear that they are both much less affected by hidden terminals than BoX-MAC due to their inherent randomness in receiver-oriented scheduling. Also, the change in optimal energy efficiency for all three MAC protocols are less than $5\%$ until the aggregate traffic rate reaches 16 packets per second. Therefore, we argue that our results for the clique network in previous sections should hold for most general networks as few deployments see a flow rate of 16 packets per second.

With hidden terminal senders, avoiding interfering concurrent traffic becomes critical for energy efficiency. Receiver centricity avoids concurrent transmissions, which substantially reduces the number of collisions in the presence of hidden terminals. In contrast, sender centricity does not explicitly randomize the transmission of different flows and relies on reliable clear channel assessment. As a result, it suffers from the increased probability of collisions when the number of concurrent flows is large in a network with hidden terminals. We expect the efficiency for sender-centric synchronous protocols such as SCP-MAC will also suffer in this case since they explicitly increase the number of contending senders by synchronizing their wakeup times.
3.6 Energy Efficiency Optimization by Duty Cycle Adaptation

We now examine how to optimize MAC performance by appropriately configuring frame lengths. Specifically, we investigate for diverse network settings the optimal frame length for MAC performance by exhaustively searching over a range of frame lengths between 10 and 5000 milliseconds with a step size of 10 milliseconds. Fig. 3.9 plots the optimal frame length versus flow rate under different network densities for O-MAC protocol. Similar trends hold for other three protocols. We observe that the optimal frame length of O-MAC is inversely proportional to its flow rate, as well as the number of flows, in a non-linear fashion. Notably, at different flow counts, the respective trends of the optimal frame length vary only slightly, suggesting that the optimal frame length can be modeled as a function of flow rate in a manner that is insensitive to the number of flows in the neighborhood. This observation motivates us to construct a purely local frame length adaptation scheme based on
Inspired by the trends of optimal frame lengths in Fig. 3.9, we introduce a simple, but nevertheless effective as we will verify shortly, frame length adaptation function for O-MAC.

\[ T_f = C_1 / r_{pkt}, \]  

(3.38)

where \( C_1 \) is a constant determined by the upper bound of frame lengths and maximal queue size, which equals 5000 milliseconds in this case; \( r_{pkt} \) represents the total incoming traffic rate destined to a receiver. Thus, to calculate the frame length at which a receiver should operate, it only requires information on the incoming traffic rate, which can be obtained either by local measurement or explicit notification from senders. In other words, each receiver may independently choose its frame length rather than rely on the aggregate traffic in the neighborhood to achieve close-to-optimal performance.
To verify the effectiveness of the proposed adaptation scheme in Eq. (3.38), Fig. 3.10 plots the loss of throughput and energy efficiency in comparison with results obtained using the optimal frame length for O-MAC. The standard deviation at each flow rate reveals the variation of performance reduction due to different flow counts that range from 1 to 20 flows. As we can observe in the figures, the delivery ratio reduction is bounded within 11% and the loss of energy efficiency is less than 13% across all tested network configurations. Note that the performance loss only appears after the MAC capacity is being reached, which is around $8 \cdot 20$ packets per second. This comparative study reveals that a local frame length adaptation results in close-to-optimal MAC performance, enabling MAC users to adapt frame lengths locally and effectively for optimizing network performance.
3.7 Discussion

In previous sections, we have shown analytical results and corroborated the validity of our modeling framework for four classes of MAC protocols. Here, we discuss promising extensions to the framework to accommodate richer network scenarios.

First, the clique network model can be extended to incorporate hidden terminals. This extension helps analyze MAC performance in general networks. In this case, the set of contenders with respect to a sender should exclude nodes that are hidden terminals. In addition, the event probabilities in Eq. (3.4) have to be modified to reflect that data reception may be corrupted by transmissions from its hidden terminals with a probability that is determined by the number of hidden terminals and their probability to transmit.

Second, we can consider other traffic patterns in the framework, such as event-based traffic, many-to-one traffic, and non-uniform traffic. For instance, it is straightforward for the framework to account for event-based traffic that is correlated in time and space, i.e., packets are generated at neighboring senders simultaneously. Since the transmission time for receiver-centric MACs relies only on the intended receiver’s
wakeup time, receiver-centric MACs are not negatively affected by the time-correlated traffic pattern. In contrast, the number of contenders (captured by parameters \( \hat{\epsilon} \) and \( \hat{\eta} \)) in sender-centric protocols increases as all senders attempt to access the channel at the same time. As for a many-to-one traffic, where packets from multiple senders are destined to a single receiver, there is little impact on sender centric MACs, since they initiate transmission without considering the wakeup schedule of the receiver. In contrast, the expected number of contenders with respect to one receiver increases for receiver-centric MACs. Now regarding a non-uniform traffic, every flow has to be treated separately. For example, flow \( i \) may take on its own rate \( r_{pkt,i} \) and frame length \( T_{fi} \). Its probability of transmitting a packet per frame becomes \( p_{ti} = \min(r_{pkt,i}T_{fi}, 1) \).

Similar modifications are required in MAC models.

Lastly, instead of evaluating a network consisting of single-hop flows, we can analyze performance for a tree-based network topology such as in Fig. 3.11. The typical communication pattern that emerges from periodic reporting is a spanning tree with traffic flowing from the leaves to the sink at the root. Periodic traffic is generated independently by leaf nodes at fixed intervals. We can detail for every node how much input traffic it is handling and traffic being sent out by neighbors in its communication range. Assume a uniform node density on the plane and a unit disk graph communication model. Every node is within a communication range consisting of a fixed number of \( c \) neighbors. Based on their hop count \( d \) to the sink node \( (d = 0) \), the output traffic rate of a \( d \)-hop node \( (r_{pkt,d}) \) can be calculated by the following formula according to [47]:

\[
r_{pkt,d} = r_{pkt,D}(D^2 - d^2 + 2d - 1)/(2d - 1),
\]

(3.39)
where $D$ and $r_{pkt,D}$ represent the maximum distance (or hop) to the sink and the flow rate at leaf node, respectively. Assume that interfering nodes on average generate the same load ($r_{pkt,d}$) as the node itself [47]. The network performance can be evaluated by iterating and combining performance results across $D$ levels. Since a relay node serves as both sender and receiver in the multi-hop setting, another flow constraint has to be applied on relay nodes, i.e., $0 \leq \psi_s + \psi_r \leq 1$, resulting in less throughput than pure sender or receiver cases.

In summary, the existing framework may be readily refined to analyze performance of MAC and upper-layer protocols in general networks.

### 3.8 Conclusions

Our comparison of four classes of MAC protocols across the configuration space yields two high-level findings. First, at high traffic loads, duty-cycled MAC protocols differ substantially in their ability to utilize channel bandwidth; in particular, the sender-centric asynchronous class (BoX-MAC) turns out to be the best candidate in terms of throughput. Second, at medium or low traffic load, duty-cycled MAC protocols achieve comparable delivery ratios but at very different energy efficiencies. Our analysis and experiments show that across a rich set of network configurations a receiver-centric synchronous protocol (O-MAC) outperforms other MAC classes in many aspects: this is because staggering receiver wakeup times and synchronizing nodes locally not only avoid generating unnecessary communication overhead (such as preambles and beacons), but also make communications more uniform and independent, which is preferable in general networks.
Our analysis of duty-cycled MAC protocols focused on a local view of the network, but our experimentation spanned both clique and non-clique networks. Our non-clique network experiments show that the performance of sender-centric asynchronous protocols degrade in non-clique networks, but importantly receiver-centric protocols perform nearly as well.

In addition, the numerical study on the optimal frame length corroborates a local but effective adaptation scheme that achieves close-to-optimal performance. We also observe that energy efficiency either increases or remains constant as traffic increases. From a cross-layer perspective, this implies that upper layers can improve efficiency by accommodating the growing traffic with higher duty cycle rather than (say) dividing the traffic over two routes.
Chapter 4: CHAMELEON: ENERGY EFFICIENCY IN MULTI-CHANNEL MAC PROTOCOL

In this chapter, we consider the energy efficiency of MAC in low-power wireless communication where multiple channels are available and the duty cycle of channel access is controllable. We show that in this setting maximization of MAC energy efficiency reduces to maximizing the aggregate channel utilization and minimizing the aggregate duty cycle channel access. Based on the reduction, we show the theoretical existence of centralized, global information protocols which achieve optimal energy efficiency in terms of channel assignment and duty cycle scheduling. Then, towards practically realizing these protocols in a distributed fashion with local information only, we present Chameleon, which assigns channels based on lightweight estimation of channel utilization and adapts the duty cycle of node reception relative to the incoming traffic. Chameleon improves energy efficiency and channel utilization not only among users internal to the network, but also in the presence of external users that share the spectrum. We compare Chameleon with state-of-the-art single-channel and multi-channel protocols. Our experimental results show substantial energy efficiency gains over these protocols, which range from an average of 24% to 66%.
4.1 Introduction

In this chapter, we consider achievable energy efficiency of duty-cycled MAC operation in networks where multiple channels (equivalently, frequencies) can be exploited. The few multi-channel protocols that have been proposed in recent years are essentially categorized into four approaches: 1) Statically partition network nodes across multiple channels so that the density of nodes on a given channel is reduced, e.g., MMSN [72] and TMCP [66]; 2) Explicitly negotiate channels to exchange data for collision avoidance based on current usage information of each channel, e.g., MMAC [59] and TMMAC [71]; 3) Migrate network nodes probabilistically at runtime from one channel to another so as to balance traffic load, using control theoretic techniques, e.g., [49] and [48]; and 4) Balance traffic load (deterministically or randomly) across multiple channels evenly so as to reduce potential interference, e.g., Y-MAC [41] for sensor networks and SSCH [16] for more general wireless networks.

All of these approaches significantly improve network goodput and, in turn, energy efficiency, in comparison with MACs that use only one single channel. Several extant protocols do not per se consider duty cycling, but we find that even if one were to include duty cycling along with these approaches, there is room for substantial improvement in goodput and energy efficiency. In the first approach, different channels are assigned to two-hop neighbors to avoid the possibility of interference; since the actual traffic is not considered, it is possible that some channels are lightly loaded and the node partitioning is thus too conservative. This approach also incurs the overhead of distributed distance-2 coloring. For the second approach, although traffic load is considered when assigning frequencies, the explicit channel negotiation
for each data communication involves nonnegligible overhead. In addition, the channel usage information has to be updated online within the distance-2 neighborhood. The third approach starts off by utilizing one channel and alleviates unfairness by probabilistically allocating a fraction of nodes into the next channel. In other words, channel utilization is expanded gradually when the goodput drops to a certain empirical threshold as measured in terms of Packet Reception Ratio or percentage of successful channel accesses. Nevertheless the goodput over the available channels is not optimized, nor is the instantaneous condition of every channel taken into account when nodes perform channel switching. As for the fourth approach, although splitting traffic loads evenly over multiple channels achieves fairness, the aggregate goodput of the network is again not necessarily maximized.

None of these approaches choose channels based on a comprehensive (albeit local) view of the current condition of all channels. Thus, the channels to which nodes are switched into may not represent the best choice. This is especially true if we take into account interference that results from the concurrent operation of external networks. Selecting channels based on a locally comprehensive yet lightweight estimator of channel utilization efficiently is the starting point for our design of a multi-channel MAC protocol, Chameleon.

Chameleon has two main components for maximizing energy efficiency. One, its multi-channel scheduler, uses the lightweight estimator to select channels in accordance with an optimality analysis presented in the following section. And second, its radio scheduler, uses a receiver-centric approach to coordinate senders and receivers

\[\text{Recent research shows that chameleons change color not to camouflage themselves but to communicate. Their “bandwidth” of communication (aka signalling) is related to the number of colors that they use. Cf.: D. Stuart-Fox and A. Moussalli, “Selection for social signalling drives the evolution of chameleon colour change”, PLoS Biol 6(1): e25, 2008.}\]
with approximately optimal efficiency; receiver-centric MACs were independently introduced in O-MAC [21] and Crankshaft [35] and shown analytically to have higher energy efficiency than sender-centric MACs [21]; more recently, the receiver-centric RI-MAC was experimentally shown to have energy gains over state-of-the-art sender-centric MACs [54, 18]. This component also realizes locally adaptive duty cycling, which staggers data communication periods so that the resulting energy efficiency is highest at the chosen duty cycle.

4.1.1 Summary of the Results

The main contributions of this chapter are as follows.

- We formalize the optimization of MAC energy efficiency in a setting where duty cycling and multiple channel utilization is possible. We show that the optimization reduces to maximizing the spectrum utilization over all available channels while minimizing the duty cycle.

- We provide a protocol that optimizes MAC energy efficiency, assuming the existence of two components, one for precisely quantifying node utilization on each channel and the other for minimizing the send-receive-idle duty cycle for a given node traffic.

- We give lightweight implementations that approximately satisfy these two components, and thus obtain the Chameleon protocol that approximates the optimal protocol. Our implementation of the first component uses a light-weight metric \( w \) which is passively computed at each receiver node. Our implementation of the second component uses a receiver-centric pseudo-random scheduling of wakeup times, so that receivers within each other interference range are unlikely to be
up simultaneously; it also chooses the receiver duty cycle to be just enough such that the receiver experiences low sender collision rate. A side-effect of this approach is that Chameleon intrinsically accommodates external interference.

- We validate, using experiments on the TelosB mote platform, that Chameleon is capable of maintaining substantially higher energy efficiency than both representative single-channel and multi-channel MAC protocols, including MMSN, Y-MAC, BoX-MAC, and O-MAC.

4.1.2 Organization of this Chapter

The rest of this chapter is organized as follows. We discuss related work in Section 4.2 and present, in Section 4.3, the system model as well as an analysis of energy efficiency optimization. We discuss a solution approach for implementing an optimal protocol and design our multi-channel protocol, Chameleon, in Section 4.4. In Section 4.5, we present experimental evaluations of relevant aspects. We make conclusions in Section 4.6.

4.2 Related Work

The state-of-the-art in research includes a significant number of multi-channel MAC protocols for sensor networks. Per our earlier classification, the first category statically assigns multiple frequencies to nodes in the network as a way of topology control, in order to reduce potential interferences. Channel allocation is carried out beforehand, and is independent of real traffic conditions, such as in [72, 66]. In [72], every node is assigned a channel for data reception such that most of two-hop neighbors do not communicate on the same channel. The TMCP protocol [66] divides
nodes into several subsets of different channels, wherein nodes only communicate within their subset for simplicity of implementation. These schemes require a centralized channel assignment algorithm to execute in the beginning and the channel utilization is not adjusted according to communication load or interference on each channel.

Another approach expands the set of channels being used when the contention on the current channel become higher than an empirically chosen threshold. A distributed protocol in [49] lets all nodes in the network start in their home channel. When the channel becomes overloaded, a fraction of the nodes migrate to the next one. Channel switching is performed with a probability such that while alleviating congestion, it avoids having all nodes jump to the new channel. However, this protocol does not have a global view of the quality of each channel, thus, channel switching need not result in higher efficiency. Another work [48] presents a centralized protocol for load balancing across channels for throughput maximization. Each node periodically decides which channel to use based on measurements from the base station. The authors assume that network throughput is optimized as long as loads are distributed equally on each channel. There are also schemes based on frequency hopping [16] which are designed mainly for wireless ad hoc networks, which involve continuous switching of channel from slot to slot even when there is no need for transmission.

Few MAC protocols explicitly design multichannel scheduling with duty cycling to achieve high energy efficiency, which is the focus of this chapter. A relatively recent multi-channel protocol, Y-MAC [41], exploits both duty cycling and multi-channel utilization. Every receiver wakes up at its non-overlap slot within each frame on home channel. If more packets need to be received, the receiver will stay awake but
hop to the next channel for reception. The merit of this scheme lies in its staggered non-overlapping channel utilization over the extended $M$ slots, while its weakness is that contiguous channel switching is expensive and the non-overlapping is guaranteed only within the $M$ slots.

### 4.3 Energy Efficiency Analysis

In this section, we first define channel utilization, spectrum utilization, and their relationship to energy efficiency. We then discuss maximization of energy efficiency for a receiver given network traffic load, in terms of expected spectrum utilization and duty cycling.

#### 4.3.1 System Model

The network consists of $N$ energy-constrained half-duplex wireless sensor nodes. Radio operation of each node is represented by a contiguous sequence of frames. Each frame consists of a number of time slots; for ease of exposition, we let this number be a global constant. Recall that a node’s *duty cycle*, implicitly over some number of frames, is the percentage of the time slots, $\psi$, when its radio is active; $\psi \in [0, 1]$. A node’s duty cycle can be further decomposed into its transmit duty cycle, the percentage of the slots when its radio is transmitting, and its receive duty cycle, the percentage of the slots when its radio is in receive or listen mode.

For a given node $i$, we refer to the packets that are sent to $i$ as its “in-traffic”, while packets that are not sent to $i$ but are overheard by $i$ or whose collision is overheard by $i$ are its “interference traffic”.

The cumulative wireless bandwidth that can be utilized by nodes denotes the network “spectrum”. Spectrum is divided into several orthogonal “channels” (or
“frequencies”) such that communications on different channels either never or only barely interfere with each other (in practice, adjacent channels are typically not completely interference free from each other [12]). Within each channel, collisions may occur if wireless devices attempt to transmit simultaneously.

The wireless network is viewed as formed by overlapping broadcast domains. Accordingly, we define a receiver’s interference set as the set of nodes whose broadcast domain covers the receiver. We let \( \eta \) denote the average size of the interference set for a given node. Let \( i, j, h \) range over nodes in the network and \( k \) range over channels of the spectrum.

With respect to a given receiver and its interference set, we define the channel utilization for a given channel, \( k \), as the ratio, \( E(k) \), of the number of time slots where a packet is successfully received to the total number of time slots. (The definition may be relativized to the number of frames considered in the definition of duty cycle.)

Consequently, spectrum utilization with respect to a receiver and its interference set denotes the overall successful transmissions among all channels over the total number of time slots normalized by the number of channels, \( M \). Hence, spectrum utilization is defined as:

\[
E_S = \frac{\sum_{k=1}^{M} E(k)}{M}.
\] (4.1)

The metric of energy efficiency being the top priority, we focus on exploring how to maximize energy efficiency at the receiver for the case of unicast traffic.

\(^8\)We note that several of our definitions are receiver-centric rather than sender-centric, as this significantly simplifies our exposition.
4.3.2 Energy Efficiency Optimization

**Problem Statement**  Given a node $i$, whose interference set is of size $\eta$, our goal is to schedule its in-traffic—i.e., choose channels and wakeup times for the $i$ and nodes sending packets to $i$—such that the resulting energy efficiency $E_E$ of $i$ is maximized.

We approach this problem by simplifying the definition of energy efficiency introduced in Eq. (1.1) for the given node $i$. First, spectrum utilization reflects the goodput resulting from communications of the nodes in the interference range of node $i$, which is $\sum \sum Z_j^i$. It follows that $\sum \sum Z_j^i = 2TME_S$, where $2ME_S$ equals the aggregate spectrum utilization and the factor of 2 reflects the benefit to both parties in a communication. The energy consumption of node $i$, which is determined by the duty cycle control scheme, is $\sum \sum (S_j^i + R_j^i) = T\sum_{j=1}^{\eta} \psi_j$. Thus, the formula below is an equivalent representation of energy efficiency.

$$E_E = \frac{2ME_S}{\sum_{j=1}^{\eta} \psi_j} \quad (4.2)$$

In order to optimize $E_E$ via maximizing $E_S$ as well as minimizing $\sum_{j=1}^{\eta} \psi_j$, the scheduler has to choose channels and wakeup times. We will first consider channel selection that maximizes the expected spectrum utilization $\hat{E}_S$, then we will discuss how to schedule the wakeup times of nodes to minimize $\sum_{j=1}^{\eta} \psi_j$.

**Maximizing $\hat{E}_S$**  Recall that $E(k)$ is the successful reception probability in the interference set of the given node. For the purpose of analysis, in this subsection, we make two assumptions. One is that the in-traffic of nodes follows a stationary process with uniform distribution of arrival times; let the in-traffic load at node $i$,
denoted by \( p_i \), be the probability that on average a packet is sent to \( i \). And two, that the node and its interference set form a clique, i.e., each of these nodes can overhear each packet sent by another of these nodes; thus if packets are concurrently sent to different nodes, collisions will result at each receiver. It follows that all nodes in the network hold the same \( E(k) \), which is defined by Eq. (4.3).

\[
E(k) = \sum_j p_j \prod_{h \neq j} (1 - p_h)
\]  

(4.3)

where \( j \) and \( h \) range over these nodes. Initially, \( E(k) \) increases as traffic loads increase. However, utilization decreases when the channel becomes overloaded, in which case collisions (or, in a contention based scheme, backoff procedures) dominate the communication.

**Lemma 1.** The expected channel utilization with respect to node \( i \), \( \hat{E}(k) \), is maximized when the aggregate traffic load in the interference set of \( i \), \( \sum_{j=1}^{\eta} p_j(k) \), increases to 1.

**Proof.** The average traffic load on channel \( k \) is \( \bar{p}(k) = \sum_{j=1}^{\eta} p_j(k)/\eta \). Hence, by Eq. (4.3), the expected channel utilization \( \hat{E}(k) = \eta \bar{p}(k)(1 - \bar{p}(k))^{\eta - 1} \). Fig. 4.1 (a) plots how \( \hat{E}(k) \) changes as \( \bar{p}(k) \) changes with interference size \( \eta \). The expected channel utilization is maximized when \( \bar{p}(k) = 1/\eta \). Since \( \bar{p}(k) = 1/\eta \) implies \( \sum_{j=1}^{\eta} p_j(k) = 1 \), it follows that maximal utilization is achieved when the aggregate load, \( \sum_{j=1}^{\eta} p_j(k) \), equals 1. Alternatively speaking, \( \hat{E}(k) \) increases as the aggregate load increases up to 1; after reaching 1, \( \hat{E}(k) \) decreases as the aggregate load increases. Hence, the total traffic load should be 1 to achieve maximal channel utilization \( \hat{E}(k) \). \( \square \)
Lemma 1 corroborates two facts: 1) the aggregate traffic load, $\sum_{j=1}^\eta p_j(k)$, is a judicious estimator of the expected channel utilization; and 2) when the estimator equals 1, channel utilization is expected to be optimum.

Now, let us consider channel selection for in-traffic $p_i$ at receiver $i$. Theorem 3 states a sufficient condition for selecting channels for load $p_i$ that maximize $\hat{E}_S$.

Let $\bar{p}_i(k)$ be the average interference load over $\eta$ on each channel $k$. Define $q(k)$ as 1 minus the current total load on channel $k$, i.e., $q(k) = 1 - \eta\bar{p}_i(k)$. Let $\vec{q}$ be the vector of $qs$ for all channels that is sorted in a nonincreasing order. Thus, $q_s$ represents the $s$-th greatest element in $\vec{q}$, corresponding to channel of index $C(q_s)$.

Let vector $\vec{\alpha} = \{\alpha(k) : k = 1, \ldots, M\}$ denote the percentages of in-traffic allocated to each channel, i.e., $\alpha(k) \cdot p_i$ is loaded on channel $k$.

**Theorem 3.** $\hat{E}_S$ is optimized if we allocate traffic load $p_i$ to channels according to fractions $\vec{\alpha}$ computed in Eq. (4.4).

$$\alpha(C(q_s)) = \begin{cases} 
q_s, & p_i - \sum_{t=1}^{s-1} q_t \geq q_s \\
\frac{p_i - \sum_{t=1}^{s-1} q_t}{p_i}, & 0 < p_i - \sum_{t=1}^{s-1} q_t < q_s \\
0, & p_i - \sum_{t=1}^{s-1} q_t \leq 0
\end{cases} \quad (4.4)$$

**Proof.** $\vec{q}$ represents residual quota of load on each channel for maximizing channel utilization according to Lemma 1. The essential idea here is to prioritize filling up
channels based on \( \vec{q} \), i.e., giving preference to those which have more residual capacity, until load \( p_i \) has been assigned completely or all \( qs \) in \( \vec{q} \) have been consumed.

Define \( \Delta p \) to be the smallest unit of load that can be assigned on a channel. Hence, load \( p_i \) consists of \( \lceil p_i / \Delta p \rceil \) units. Before adding a unit \( \Delta p \) into channel \( k \), the expected utilization on channel \( k \) is \( \hat{E}(k) = \eta \bar{p}_I(k) (1 - \bar{p}_I(k))^{\eta - 1} \). After adding \( \Delta p \), by Eq. (4.3), the expected utilization becomes \( \hat{E}'(k) = \eta \bar{p}_I(k) (1 - \Delta p) (1 - \bar{p}_I(k))^{\eta - 1} + \Delta p (1 - \bar{p}_I(k))^\eta \). Thus, the utilization gain, \( \Delta \hat{E}(k) \), on channel \( k \) after appending each \( \Delta p \) would be

\[
\Delta \hat{E}(k) = \Delta p (1 - (\eta + 1) \bar{p}_I(k)) (1 - \bar{p}_I(k))^{\eta - 1},
\]

which is a monotone decreasing function of \( \bar{p}_I(k) \). The smaller the \( \bar{p}_I(k) \), the higher the utilization gain will be. Since \( \Delta p \) is an atomic unit, assigning the channel with lowest \( \bar{p}_I(k) \) will provide the highest \( \Delta \hat{E}_S \), where \( \Delta \hat{E}_S = \Delta \hat{E}(k) \) and \( k \) is the channel assigned to the \( \Delta p \) load.

Ideally, all \( \lceil p_i / \Delta p \rceil \) units would be added into the channel with the lowest \( \bar{p}_I(k) \) to maximize total utilization gain. According to Lemma 1, however, the total load on each channel \( k \) should not exceed 1 to achieve maximal utilization. \( C(q_s) \) denotes the channel which has \( s \)-th smallest \( \bar{p}_I(k) \) and \( q_s / \Delta p \) is the number of units that can be added to a given channel before exceeding the maximum. Therefore, sequentially filling up each channel according to the order in \( \vec{q} \) will maximize total \( \hat{E}_S \).

Consider the assignment of load to channel \( C(q_s) \). The number of load units of \( p_i \) that are yet to be assigned is \( (p_i - \sum_{t=1}^{s-1} q_t) / \Delta p \). If this number is non-positive, indicating that all units of \( p_i \) have been assigned to channels earlier in the order of \( \vec{q} \), the fraction assigned to channel \( \alpha(C(q_s)) \) is 0. Otherwise, if the number of unassigned load units is less than \( q_s \), we can assign all of \( (p_i - \sum_{t=1}^{s-1} q_t) / \Delta p \) units to channel \( C(q_s) \).
\[ \alpha(C(q_s)) = (p_i - \sum_{t=1}^{s-1} q_t)/p_i \] in this case. If the number of unassigned units is not less than \( q_s \), we can fill up this channel with \( \alpha(C(q_s)) = q_s/p_i \).

In essence, Theorem 3 yields one approach for optimizing the spectrum utilization by choosing channels for the in-traffic at a node.

**Minimizing \( \psi \)** We now consider scheduling for duty cycle minimization. It is straightforward to show a “centralized TDMA and duty cycling” scheduler that has full information of the arrival times of all packets would suffice to this end. This scheduler (having scheduled the existing traffic in the network) can schedule packet communication time so that no collisions happen, as well as senders and receivers are scheduled to wakeup exactly at these times. Lemma 2 states that nodes running the duty-cycled TDMA will minimize gross duty cycle \( \sum_{j=1}^{\eta} \psi_j \).

**Lemma 2.** Given traffic load \( p_i \) and the arrival time of the in-traffic of \( i \), the centralized TDMA and duty cycling scheduler minimizes the total duty cycle \( \sum_{j=1}^{\eta} \psi_j \).

**Proof.** Duty cycles of nodes that are neither senders nor receivers of packets in the in-traffic of \( i \) will remain unchanged. As for nodes involved in the traffic, the scheduler trivially minimizes the wakeup times, since there are no superfluous sends or receives or idle slots. The total duty cycle consumed by the load \( p_i \) is minimized to be twice of the load, i.e., \( 2p_i \).

### 4.4 Energy Efficient Multi-Channel Protocol Design

In this section, we present our energy efficient multi-channel access protocol, Chameleon. First, guided by Theorem 3 and Lemma 1, we present the component that (re)assigns channels to load units. Then, we design a light-weight, local,
receiver-centric scheduler that approximates the heavy-weight centralized scheduler indicated in Lemma 2. We conclude with an overview of our TinyOS implementation of Chameleon.

4.4.1 Channel to Load Assignment

In Lemma 1, channel utilization is estimated via the sum of traffic loads, including in-traffic load and interference load. These two loads also determine channel assignment according to Theorem 3. Basically, each node, say \( i \), continually performs three tasks: (i) determines the in-traffic for \( i \), (ii) determines the interference traffic to \( i \), and (iii) chooses channels for the in-traffic according to Eq. (4.4).

For task (i), the in-traffic load at node \( i \) (i.e., the exact instantaneous \( p_i \) value) can be computed either by appending rate information to data packets sent to receiver \( i \) or by locally calculating the rate of incoming load at \( i \); we chose the former.

Task (ii) involves collecting information about the interference load at node \( i \), \( \eta \bar{p}_I(k) \) for each channel \( k \). Rather than let \( i \) actively coordinate with all nodes in its interference set to compute the value, we introduce a local interference estimator for \( \eta \bar{p}_I(k) \) in the next subsection.

Local Interference Estimator

Let interference estimator \( I(k) \), defined in Eq. (4.6), refer to the probability that some interferers of node \( i \) transmit on channel \( k \).

\[
I(k) = 1 - (1 - \bar{p}_I(k))^\eta
\]

(4.6)

It follows that \( \eta \bar{p}_I(k) \) is estimated by the exponential function of \( I(k) \), i.e., \( e^{I(k)} \). We leverage the similarity between the sum of traffic loads, notated by \( \sum_{j=1}^{\eta} p_j(k) \),
and $p_i(k) + e^{I(k)}$, denoted by $w(k)$. Hence, $w(k)$ is employed to compute channel utilization.

Fig. 4.1 (b) shows an instance of the relation between $\sum p_j(k)$ and $w(k)$. We consider a clique network wherein six pairs of nodes communicate independently on the same channel, each with an arbitrarily chosen traffic load in the range $[0,1]$. Each receiver locally computes the metric $w(k)$, where $k$ is fixed. Fig. 4.1 (b) plots the mean value $w(k)$ and the standard deviation of the six receivers versus the aggregate traffic load $\sum p_j(k)$. Here, the same value of the aggregate load corresponds to a few different sequences of traffic loads $\vec{p}$. We observe in the figure that $w(k)$ is approximately linear with $\sum p_j(k)$, which verifies that $e^{I(k)}$ is a feasible estimator for interference load, in lieu of the metric $\eta \vec{p}_j(k)$. Additionally, the locally computed deviation of the $w(k)$s is very small, i.e., the average standard deviation shown in the figure is around 0.005.

Another relevant observation from our analysis is that when $\sum p_j(k)$ equals 1, $w(k)$ is equal to 2 (see the figure). This is the state where $\hat{E}(k)$ is optimized, and we refer to it as $w^*$. Moreover, when parameter $\eta > 2$ and $e^{I(k)} \leq w^*$, the linear relation between $\sum p_j(k)$ and $w(k)$ is preserved for different configurations of $\eta$. It follows that metric $q(k)$ in Theorem 3 may be substituted by the local metric $w^* - e^{I(k)}$ as we perform task (iii).

In particular, the computation of $I(k)$ does not involve sending any specific information, in contrast with the Channel Access Ratio message used in many multi-channel protocols, such as [49]. We explain how interference level $I(k)$ is measured in the next subsection, and how the local metric is used in task (iii) in the following subsection.
Figure 4.2: Mean error of measured $I(k)$ versus the duty cycle of measurement.

**Estimator Implementation**

The value of $I(k)$ is measured passively at node $i$ by randomly listening to channel $k$ when $i$ is not performing data reception or data transmission. Measurement is performed periodically (at a low duty cycle). For each period, the ratio of the number of busy slots to the total number of checked slots yields the value for $I(k)$.

In terms of implementation, we let nodes perform a continuous Clear Channel Assessment (CCA) check on a given channel during each check slot to determine whether that slot has interference traffic or not. (For the TelosB platform, we empirically chose the channel monitor slot length to be 3ms.) Due to the inefficiency of float operation in the mainstream sensor platforms, we normalize and quantize load into integer “levels”. We let the unit of load, $\delta p$, be 0.01; 0.01 thus corresponds to the integer level 1. Traffic $p_i$ and interference $I(k)$ are normalized to $[p_i/\delta p]$ and $[I(k)/\delta p]$, respectively. Furthermore, we pre-compute the corresponding value of $e^{I(k)}$ for each level of $I(k)$, thus every receiver maintains a vector $\exp I = \{e^{I(k)} : k = 1, 2, ..., M\}$, representing the interference traffic load on each channel.

The choice of measurement period involves a tradeoff between accuracy and energy consumption. To understand this tradeoff, we conducted experiments in which all 5
nodes transmit independently at a specified rate. Each experiment was repeated for traffic loads of 0.01 (approximately 1 packet per second), 0.05, and 0.1, respectively, and also with the nodes performing channel measurement at different duty cycles. We let channel monitoring be triggered by a randomized timer that fires between $0.5T$ and $1.5T$, where $T = 15s$. When the timer fires, the node monitors the channel for several slots if the radio is not being occupied; otherwise, it waits to measure until the radio is released by other processes. The cumulatively measured value of $I(k)$ is reported at a fixed interval of every 5 minutes. To further reduce error, a weighted moving average to consecutive measurements is computed. Hence, $I(k) = \alpha I(k) + (1 - \alpha)I'(k)$, where $I'(k)$ is the value of last measurement. We let $\alpha$ be 0.6 in our experiments. After each report, the counter of $I(k)$ is cleared to zero and another period of monitoring started.

Fig. 4.2 plots the mean error between the measured level and the expected value with the monitoring channel at different duty cycles ranging from 0.01% to 2%. The x axis represents the duty cycle of passive channel monitoring. Initially, as the monitoring duty cycle increases, the precision of measure increases significantly; however, the improvement reduces when duty cycle is greater than 0.2%. The corresponding average error is at a level of 1 to 2. Thus, to update channel interference level $I(k)$ at an interval of 5 minutes, a duty cycle of 0.1% to 0.2% for channel measurement seems adequate. Alternatively, checking randomly every 200 slots would provide an acceptable measurement for a channel (recall that each slot is 3 ms).

Chameleon offers upper layers the option to adapt channel update interval from time to time to deal with dynamic environments. In the following experiments, we use a 0.2% duty cycle for interference monitoring, unless stated otherwise.
Algorithm for Channel to Load Assignment

Having obtained in-traffic and interference load, task (iii) is implemented by Algorithm 1. Given normalized levels of \( p_i \) and \( \vec{e} I \), we first compute the number of acceptable units on each channel, in \( \vec{q} \) (lines 1 to 7). Lines 17 to 26 assign units to each channel according to Theorem 3, which results in a vector \( \vec{V} \) of size \( M \), e.g., \( \vec{V} = (3, 7, 1, 0, ..., 0) \), where each element represents the units allocated to the channel. Thus, \( p_i \) is split across the channels in proportion to \( \vec{V} \). (Which channels to use in which frame is discussed later in this section.) If the sum of the available capacity, \( \sum_{k=1}^{M} q(k) \), is less than the total \( p_i \), cf. line 9, senders are notified to reduce their outgoing traffic if possible.

Algorithm 1 Channel to Load Assignment

<table>
<thead>
<tr>
<th>Require: ( p_i, \exp I )</th>
<th>1: for ( k = 1 ) to ( M ) do</th>
</tr>
</thead>
<tbody>
<tr>
<td>2: if ( w^* - e^I(k) \leq 0 ) then</td>
<td></td>
</tr>
<tr>
<td>3: ( q(k) \leftarrow 0; )</td>
<td></td>
</tr>
<tr>
<td>4: else</td>
<td></td>
</tr>
<tr>
<td>5: ( q(k) \leftarrow w^* - e^I(k); )</td>
<td></td>
</tr>
<tr>
<td>6: end if</td>
<td></td>
</tr>
<tr>
<td>7: end for</td>
<td></td>
</tr>
<tr>
<td>8:</td>
<td></td>
</tr>
<tr>
<td>9: if ( \sum_{k=1}^{M} q(k) &lt; p_i ) then</td>
<td></td>
</tr>
<tr>
<td>10: Inform senders (optional);</td>
<td></td>
</tr>
<tr>
<td>11: end if</td>
<td></td>
</tr>
<tr>
<td>12:</td>
<td></td>
</tr>
<tr>
<td>13:</td>
<td></td>
</tr>
<tr>
<td>14: Sort ( \vec{q} ) in non-increasing order</td>
<td></td>
</tr>
<tr>
<td>15: ( \vec{q} = (q_1, q_2, ..., q_M) )</td>
<td></td>
</tr>
<tr>
<td>16: the channel index of ( q_s ) is ( C(q_s) );</td>
<td></td>
</tr>
<tr>
<td>17: for ( s = 1 ) to ( M ) do</td>
<td></td>
</tr>
<tr>
<td>18: if ( p_i - \sum_{t=1}^{s-1} q_t \geq q_s ) then</td>
<td></td>
</tr>
<tr>
<td>19: ( V(C(q_s)) \leftarrow q_s; )</td>
<td></td>
</tr>
<tr>
<td>20: else</td>
<td></td>
</tr>
<tr>
<td>21: if ( p_i - \sum_{t=1}^{s-1} q_t &lt; q_s ) then</td>
<td></td>
</tr>
<tr>
<td>22: ( V(C(q_s)) \leftarrow p_i - \sum_{t=1}^{s-1} q_t; )</td>
<td></td>
</tr>
<tr>
<td>23: else</td>
<td></td>
</tr>
<tr>
<td>24: ( V(C(q_s)) \leftarrow 0; )</td>
<td></td>
</tr>
<tr>
<td>25: end if</td>
<td></td>
</tr>
<tr>
<td>26: end for</td>
<td></td>
</tr>
</tbody>
</table>

Each node starts with conducting a cumulative measurement for every available channel, followed by independently allocating its load to the corresponding channels. As network load varies, the channel monitoring daemon updates channel assignment.
(in vector $\vec{V}$) at each receiver. To alleviate fluctuations caused by simultaneously channel switching, every receiver carries out its channel reassignment with a random interval.

### 4.4.2 Receiver-Centric Wakeup and Channel Scheduler

Lemma 2 indicates that there exists in theory a centralized, global information scheduler for maximizing energy efficiency. The scheduler continually performs for each node, say $i$, the following task: it computes the time at which each in-packet at $i$ is sent without interfering with any of the packets scheduled thus far; it also updates the sleep-wakeup schedule of the nodes so that they wake up only when they are involved in transmitting or receiving each in-packet to $i$. Note that the packet transmission time scheduling yields an in-traffic whose arrival time may no longer satisfy a uniform distribution, which we assumed in the analysis shown in Section 2, but since this scheduler enforces collision freedom, the expected $E_S$ and $E_E$ are not negatively affected. However, this centralized scheduler is of high complexity.

**Wakeup Scheduler**

We now discuss a distributed, light-weight component that efficiently approximates the centralized scheduler. Specifically, we adopt a synchronous, receiver-centric scheduling approach that locally avoid collision and schedules sleep-wakeup. This approach is exemplified by receiver-centric synchronous MAC class such as O-MAC [21] and Crankshaft [35], which has been shown in Chapter 2 and 3 to best approximate the optimal scheduler with duty cycling.

The basic idea that we borrow from receiver-centric synchronous MACs is this: Each receiver has a pseudo-random scheduler which determines its wakeup slots. The
wakeup schedule is advertised to neighbors, compactly since essentially the pseudo-
random seed needs to be shared, via a neighbor discovery process. When a node
discovers this receiver, it also obtains this receiver’s state (of pseudorandom gener-
ation), and thus the node can generate the receiver’s wakeup schedule. When the
node wishes to send to the receiver, it wakes up at the next slot at which the receiver
will be awake and attempts to communicate. Two basic modules, neighbor discovery
and time synchronization, are used and in turn the module offers Send and Receive
interfaces.

Chameleon adopts these basic interfaces from those in O-MAC. This decentralized
pseudo-random scheduling staggers nodes’ wakeup times with high probability, and
has been proven [21] to utilize less duty cycle (i.e., to have higher energy efficiency)
under the same traffic load in network than other sender-centric protocols, such as
B-MAC, BoX-MAC [54], X-MAC [18] and others. As compared to asynchronous
receiver-centric protocols, such as RI-MAC, a sender in O-MAC will not wake up for
an average of half a frame waiting for beacon from its receiver, thus the duty cycle at
sender side of O-MAC is obviously less than that of RI-MAC although the receiver’s
duty cycles are comparable in two protocols.

In other words, in Chameleon’s receiver-centric scheme, the senders’ wakeup times
are implicitly scheduled. Since receivers wakeup at random times in each frame, the
likelihood that two interfering receivers will simultaneously receive is low. In O-MAC,
a short beacon is broadcast by the receiver as it wakes up to compensate for slot
misalignment with potential senders. The beacon contains an adaptive contention
window size determined by the receiver side for collision avoidance, based on the
expected number of concurrent senders for that receiver. O-MAC is also flexible in
adapting duty cycle to incoming traffic load. A sender is allowed to continuously send queued packets to a receiver as long as the sender grabs the channel for the first packet. When a node fails in competition, it continues to compete for the next frame.

Channel Scheduler

We extend the basic O-MAC scheduler in two ways: 1) channel association with frames; 2) channel notification from receivers to senders.

First, the scheduler associates a channel with each frame. This channel is used by the receiver in all slots in which it wakes up during that frame. We implement this association using a vector of units assigned to each channel, \( \vec{V} \), which has size \( M \). Given an assignment \( \vec{V} \), the receiver maintains a shadow copy \( \vec{V}' \), which is initially set to \( \vec{V} \). In each frame, it checks the next \( k \) in \( \vec{V}' \). If the value of \( V(k) > 0 \), then channel \( k \) is used in the next frame and the current value in \( \vec{V}' \) is decremented; otherwise, the next channel is checked until all values become 0. Then, \( \vec{V} \) is copied to \( \vec{V}' \) again and the above procedure repeated. In this way, nodes use multiple channels in proportion to \( \vec{V} \). Equivalently speaking, the incoming traffic to the node is split over multiple channels.

Second, there are two ways in which the receiver shares its updated channel assignment with senders in the receiver-centric approach: asynchronously, through the neighbor discovery process and, synchronously, through beaconing in the first wakeup slot at the beginning of each frame. In the former case, nodes independently compute each other’s wakeup slot and channel. The updated channel-wakeup schedule \( V \) has to be notified to neighbors via the discovery module within certain amount of time. Each sender keeps its own updated \( \vec{V}' \) and the current index of the receiver, generating future wakeup slots and channels independently. This scheme is realized
by leveraging the asynchronous neighbor discovery protocol, Disco [27], which schedules radio wake times at multiples of prime numbers, ensuring deterministic pairwise discovery and rendezvous latency. Disco operates on a well known channel, called the home channel. We add several small pieces of information to the packets sent out by Disco, related to time synchronization, channel assignment, and wakeup schedule. When a receiver starts to change its channel assignment schedule, it may accelerate propagating a channel update, by increasing the duty cycle of Disco. After exceeding the deterministic rendezvous period, Disco goes back to previous low duty cycle. The energy cost of updating schedules through Disco is nontrivial, especially when frequent updates exist. Moreover, the discovery schedule may interrupt with node’s listen schedule more frequently in this case.

In the latter case, status is updated by advertising the receiver’s current channel at the beginning of each frame, using the home channel. Senders do not maintain any channel information, instead they listen to the home channel during the wakeup slot of receiver. The receiver broadcasts the channel it is going to use for current
frame in a short beacon on the home channel. Note that the beaconing is part of O-MAC protocol. Following this beacon, potential senders and receiver all switch to the chosen channel for the rest of communication. Specifically, receiver switches to the chosen channel after sending out beacon and senders change to the channel after receiving the beacon. The total beaconing and channel switch time is approximately 5 ms on the TelosB platform.

4.4.3 Implementation

We implemented Chameleon in TinyOS 2.x for the CC2420 radio platform, which is a packetizing radio used in popular TelosB and MicaZ motes; the code is readily ported to motes with streaming radios such as the CC1000. The composition of Chameleon is shown in Fig. 4.3.

The Scheduler module in Fig. 4.3 includes three basic modules provided by O-MAC: listener, sender, and discovery & synchronization. The Listener module decides node wakeup times and durations, while Sender determines when to transmit application packets given the state maintained in the neighborhood table. The Discovery & Sync module performs relative slot synchronization (with a modified FTSP protocol) on the basis of asynchronous discovery (with Disco); these processes have rather low overhead.

The ChannelMonitor module realizes the bulk of the functionality of Chameleon, including the periodic measurement of $I(k)$ and the channel selection. It implements and provides the interface RadioControl for the purpose of transparently performing channel monitoring task, giving higher priority to O-MAC tasks with the radio resource. Chameleon only uses the radio when O-MAC is not occupying the resource.
Whenever O-MAC attempts to start the radio, Chameleon immediately stops its monitoring task and returns the control of radio to O-MAC. ChannelMonitor also generates the channel schedule, which is input to the Listener module which implements the desired channel switching upon wakeup. In the diagram, colored components represent Chameleon modules which are modified or new with respect to the original O-MAC protocol. The Sender module is also modified to incorporate multichannel feature when transmission. The interfaces provided by Chameleon are MCM (Multi-Channel Monitoring) as follows.

```c
interface MCM {
    command error_t start ();
    command error_t stop ();
    command uint8_t getCh ();
    command void setCh (k);
    event void setChDone (error);
    command ch_arr chVector ();
    command void setUpdateInterval (t); }
```

Command `getCh` returns the index of the channel to use for communication based on recent channel monitoring result. The returned value of this command is included in the beacon sent out when the receiver wakes up. Command `setCh` is called to switch the channel for data transmission and the `setChDone` event is signaled after radio has stabilized on the new channel. `chVector` returns the current channel allocation in an array as $V$, while command `setUpdateInterval` provides a way for the application to adjust the update interval of channel assignment.
4.5 Protocol Evaluation

We evaluated Chameleon via both simulations, in Matlab, and experiments, based on an implementation in TinyOS 2.x for the TelosB platform [6]. We show, using simulation [50], that the performance of not only the metric $w$ but also Chameleon compares favorably with other multi-channel MAC protocols under various traffic scenarios and network topologies. To validate Chameleon’s performance in the presence of a realistic environment and (TelosB) platform effects, we experimentally evaluated three main metrics, namely, the end-to-end delivery ratio, the average receive duty cycle, and energy efficiency, of Chameleon with other benchmark protocols under various circumstances.

Delivery ratio is computed periodically, i.e., the number of successfully received packets at destinations divided by the number of packets attempted to be sent from sources. Due to the receiver-centric nature of these multi-channel protocols, we only consider the receive duty cycle at a node, which is represented by the fraction of active periods for listen or receive to the total period of time. The transmit duty cycle is approximately equal to the receive duty cycle because both Y-MAC and Chameleon are synchronous protocols. Given that data period of each slot takes $t$ time, the energy efficiency is $t$ multiplied by the number of slots that received packets successfully divided by the total active time for listening or receiving. We likewise corroborated its ability to tolerate external traffic and its relative improvements over both single channel (specifically, BoX-MAC and O-MAC) and multi-channel protocols (MMSN and Y-MAC).

Towards comparing with the other two multi-channel protocols in a fair manner, we made several necessary modifications to MMSN [72]. The frequency assignment of
MMSN evenly allocates the available channels to neighbors. For media access, MMSN as specified does not consider duty cycling. As in Chameleon, however, we let each receiver listen at its own slot, and thus avoid the more expensive frequency toggle preamble incurred in the original specification of MMSN, given that senders are aware of receiver schedule. In other words, the modified version of MMSN that implemented has reduced protocol overhead. Given that the frequency used by a node is predefined and known to neighbors, the protocol overhead of this modified version of MMSN is close to zero. We implemented the modified MMSN and Y-MAC on TelosB platform. Based on current implementation of O-MAC, the average slot length of Chameleon is 16ms (same as O-MAC) and of Y-MAC is 20ms; the latter is larger since channel switching (and synchronization) is performed in every slot. MMSN operates at full duty cycle as in its original specification. The data packet size is fixed at 60 bytes. All data communications are performed in unicast mode. The size of the backoff window in each slot is 4ms. Neighbor discovery and time synchronization services are provided by O-MAC. In the comparison, we did not let Chameleon enforce restrictions on incoming traffic even if all channel capacities had been exceeded. (Such policing would, however, help the performance of Chameleon.) The monitoring overhead is zero for both MMSN and Y-MAC since channel assignment is done either a priori or deterministically; and around 0.6% duty cycle for Chameleon under three channels.

As for the single channel protocols, we used existing implementation of BoX-MAC, which is representative of duty-cycled asynchronous protocols, and O-MAC, which is representative of duty-cycled synchronous protocols. BoX-MAC [54] is the default low power listening protocol implemented in TinyOS-2.x. We let its receive check interval be set to 100ms.
Metrics for a Clique Network

Our first experiment was repeated for the five protocols in a clique network whose traffic load increases over time. The load increases adding independent flows to the network, with no flow sharing a source or a destination node with any other flow. Flows have one of three rates, with 1 packet every 100 milliseconds or 50 milliseconds, or 25 milliseconds, resulting in a load of approximately 10%, 20%, or 40% duty cycle, respectively. 6 independent flows are successively added in the network, with loads of 10%, 20%, 40%, 40%, 10%, and 20% respectively.

To avoid experimental error due to external interference from the environment, we collected measurements on the noise level for every available channel in our testbed. This gave us three relatively free channels in our testbed for this experiment, i.e., channels 22, 24, and 18. (Note that although there are 16 channels available on TelosB platform, it has been shown that adjacent channels actually interfere with each other [39]. Therefore, we avoided using adjacent channels in all our experiments.)

Fig. 4.4 (a)(b)(c) plot the metrics for these five protocols. We see that single channel protocols are much more negatively affected by the augmentation of traffic load than are multi-channel protocols. The packet delivery ratio of O-MAC is only slightly higher than that of BoX-MAC, but the duty cycle of BoX-MAC is 2 to 4 fold of O-MAC, suggesting that synchronous receiver-centric MAC protocol may be substantially more energy-efficient than asynchronous sender-centric protocols. The efficiency of both protocols decreases significantly as the traffic load increases. We also see that the overhead involved in Chameleon over O-MAC is within a 1% duty cycle.
Chameleon maintains the highest delivery ratio of the three multichannel protocols as the traffic loads increases. In comparison with Y-MAC, MMSN has a worse delivery ratio because channel 22 is overloaded with flows (3 receivers are statically assigned to the same channel). Fig. 4.4 (b) shows the average duty cycle of the receiver, which is proportional to the average traffic load. Y-MAC incurs about 10% higher overhead than Chameleon due to its continuous channel switching scheme. On the other hand, the primary overhead of Chameleon—channel monitoring—involves insignificant energy consumption. Fig. 4.4 (c) illustrates the overall energy efficiency of each protocol. Chameleon has 62% to 55% efficiency as internal network load grows, which is on average 40% and 20% more efficient than modified MMSN and Y-MAC, respectively.

**Metrics for a Clique Network with External Interference**

Static assignment of load to channels, as in Y-MAC and MMSN, is inherently inefficient if the utilization of the shared spectrum by external systems is not monitored. Since Chameleon monitors channels comprehensively, it is intrinsically adaptable to dynamic and unknown wireless environments. Our next experiment introduced an external interferer to the network. In this experiment, 3 flows with duty cycles of 10%, 20%, and 40% exist in the network, and they use 3 of available channels. Time is divided into 8 periods. In period 1, there is no external interferer. During times 2 to 4, the interferer transmits on channel 18 with loads of 20%, 40%, and 60% sequentially. Later, interferers switch to channel 22 at time 5 and repeat the same increasing load pattern on channel 22.

Fig. 4.4 (d)(e)(f) shows the resulting delivery ratio, mean receive duty cycle, and energy efficiency. Initially, Chameleon and MMSN both distribute three flows into
Figure 4.4: The number of internal network flows increases in an experimental clique network, (a) average packet delivery ratio, (b) average duty cycle, (c) energy efficiency. External interference load changes in an experimental clique network (d) average packet delivery ratio, (e) average duty cycle, (f) energy efficiency.
the three channels while Y-MAC evenly allocates traffic onto every channel. When the interferer on channel 18 increases its load, both MMSN and Y-MAC retain their current channel usage resulting in a reduced delivery ratio. In contrast, Chameleon detects the interference level increase on channel 18 and moves its traffic to other better channels. Thus, a high packet reception ratio as well as high energy efficiency is maintained by Chameleon’s channel allocation scheme.

4.6 Conclusions

This chapter presented a new multi-channel MAC protocol, Chameleon, for duty-cycled wireless sensor networks. Chameleon betters the energy efficiency of existing protocols by adapting both the duty cycle and the channels that are being used. On one hand, it attempts to maximize spectrum utilization, via a light-weight channel utilization metric $w$ that lets it split loads across channels effectively. On the other hand, it uses a receiver-centric approach to minimize on-duty time at the receiver, while letting senders wakeup only when they need to send and know the receiver is awake.

Experimental results confirm that Chameleon enhances energy efficiency substantially as compared to other multi-channel protocols under various internal traffic scenarios. Related experiments have shown us that external interference in long-lived WSNs is nontrivial, and is also typically unpredictable. Chameleon naturally coexists with dynamic conditions in spectrum and improves energy efficiency to a large extent.

The current design of Chameleon has not involved the broadcast scenario, which will be extended in the future. Future work will also examine the dynamics of
Chameleon under different network topologies. We seek to address potential stabilization issues in channel selection via lightweight coordination among receivers.
Chapter 5: THERMONET: FINE-GRAIN ASSESSMENT OF BUILDING COMFORT AND EFFICIENCY

One emerging market for smart sensors and control systems is to embed sensor networks into indoor environment for monitoring and actuation. Understanding the performance of the HVAC system in large buildings is a prerequisite for optimizing their energy efficiency. Fine grain performance analysis has not, to our knowledge, received adequate attention thus far. To address this issue we evaluate the thermal comfort and the energy efficiency of a relatively modern HVAC system in a large building based on building-wide high-fidelity environmental data collected via a wireless sensor network over 12 months. Access to fine grain information reveals temporal and spatial dynamics that help quantify the level of (non-)compliance with the system’s control objective and the building’s thermal comfort standards: we find over-conditioning at multiple time scales which offers opportunities for reduced operating cost, and identify building anomalies and ill-conditioned rooms that need maintenance. This chapter moreover presents a data-driven thermal model that is useful in improving control strategies of the building.
5.1 Introduction

Given that approximately 35% of the energy in the United States is used for heating, ventilation, and air-conditioning systems [29], a number of efforts have been made over the past decade to improve HVAC performance by instrumenting wireless sensor networks in residences, campus offices, and production data centers [29, 11, 52, 24]. A major theme of this research has been the leveraging of occupancy information to optimize HVAC schedules or adapt setbacks. Typically, a per-room controller is assumed or else the performance of a building-wide HVAC system is simulated in tools such as eQuest [2] and EnergyPlus [1]. However, in modern buildings accommodating hundreds if not thousands of rooms, simulations tend to be insufficient for HVAC performance evaluation given the complexity of thermal and air dynamics in large spaces. Several factors underlie this complexity, including interdependencies in the design of air distribution/diffusion, customization of air handling to the respective functions of diverse rooms, and selection of HVAC control policies. We are therefore motivated to consider fine grain instrumentation and evaluation of an existing HVAC system, to provide not only realtime feedback but also long-term data of the environment that can be used for optimizing building control schedules.

Specifically, our study analyzes HVAC performance in terms of thermal comfort versus energy consumption based on dense, long-lived environmental monitoring of a relatively large building, the 9-floor new Dreese Laboratory\(^9\) at the Ohio State University, which was added to an existing structure in 1994. Our conclusions are based on year-long data from a 100+ node sensor network, which reports temperature and light data from almost all rooms in the building once every 15 minutes. The building,

\(^9\)In the rest of the chapter, new Dreese Laboratory is abbreviated as Dreese.
Figure 5.1: Picture of (a) Dreese Laboratory at OSU and (b) ThermoNet architecture illustrated for the 2nd floor of Dreese Laboratory.

pictured in Fig. 5.1 (a), houses the Computer Science & Engineering Department in more than 100 rooms, which serve as offices, labs, conference rooms, classrooms, and a compute center.

Since it came into use, there have been many complaints of uncomfortable room temperatures from Dreese occupants. According to a recent survey, only 34.3% of occupants in Dreese consider their rooms to have been mostly comfortable over the last year. In contrast to BACnet\textsuperscript{10}-enabled buildings, Dreese rooms are not monitored by environmental sensors; thus, building operators do not have access to realtime or historical data for each room, which makes it difficult for them to identify or troubleshoot HVAC problems. On the other hand, installing, wall-powering, and wiring instruments (such as temperature, humidity, and light sensors) in every room involve a significant amount of work and expense, especially for legacy buildings, since

\textsuperscript{10}The BACnet protocol defines a number of services that are used to communicate between building devices, which allows communication of building automation and control systems for applications such as HVAC control and lighting control and their associated equipment.
interfacing with the existing building management system is cumbersome and costly. To tackle this problem, this chapter presents a promising low-cost WSN solution, ThermoNet, which enables per-room, long-lived environmental monitoring of Dreese.

We begin with a description of the control objective and the architecture of the HVAC system in question. The current target is to maintain the whole building all through the year at 72°F with ±2 degree error during the daytime, and between 60°F and 80°F at night. The architecture includes two standard Variable Air Volume (VAV) Air Handler Units (AHUs), which respectively reset their discharge air set points between 55°F and 65°F, by periodically comparing a reference set point with the highest of the temperatures sampled on-line from three pre-selected rooms. The two AHUs are shut down between 11:30pm at night and 5:30am/7:00am respectively in the morning. A second tier of independent (pneumatic) control exists in approximately half of the rooms, whereby the room can control its discharge set point with a change in air velocity and a reheat coil. However, the local controller is covered by a metal lid in the room so that it is accessible only to building maintainers instead of room occupants. Occupants in Dreese rooms are not allowed to open or adjust the controller; thus, they have little control over room temperature set points.

5.1.1 Summary of the Results

It is known that thermal comfort is a complex measurement that depends on many aspects. The most common comfort measurement is Fanger’s Predicted Mean Vote (PMV), which as standardized in ISO 7730 [56], depends upon air temperature, radiant temperature, humidity, air velocity, occupants clothing and activity. Measuring PMV is rather involved, i.e., radiant temperature and air velocity sensors tend to
be expensive and complex, as a result, large scale installations would be expensive. Given this impediment, our work focuses on evaluating temperature with respect to the HVAC control objective rather than attempting to measure PMV. Broadly speaking, as detailed in the following sections, our contributions and findings regarding the building’s HVAC operation are as follows.

- We design and implement a low-cost, large-scale WSN that enables fine grain evaluation of HVAC performance in legacy buildings while meeting a multiyear-long lifetime requirement via duty cycling and adaptive power control.

- With respect to the control objective of HVAC system, although the building-wide average temperature is typically within the targeted limits, the comfortable area averages at only 47% of the building. There are not only a significant number of persistently ill-conditioned rooms but also a significant number of intermittently ill-conditioned rooms that warrant maintenance for thermal comfort.

- A substantial opportunity for energy savings exists based on dynamics at different time scales: (i) Many rooms are overcooled (overheated, respectively) in summer (in winter, respectively). (ii) During season changes, switches in the cooling and heating modes can yield overcooling and overheating respectively. (iii) Daily patterns of building temperature and (illumination-based) occupancy also show that the AHUs can be shut for several more hours at night while still meeting the control objective.

- We present a data-driven thermal model that offers the aggregate temperature of the building using a small set of room temperatures.
5.1.2 Organization of this Chapter

Related work on HVAC system evaluation is reviewed in Section 5.2. Section 5.3 introduces the infrastructure and protocol design of ThermoNet. Section 5.4 investigates the comfort level of the building and Section 5.5 illustrates opportunities in optimizing energy efficiency of the HVAC system. A thermal model based on long-term sensing measurement is discussed in Section 5.6. Section 5.7 makes concluding remarks.

5.2 Related Work

Research has shown that lack of visibility into building operating conditions is a root cause for low energy efficiency. Over the last decade there has been a growing consensus that low cost WSNs yield appropriate solutions for collecting high fidelity data about the environment. RACNet [52] was developed with this motivation and is perhaps the work most related to ours. Towards understanding the thermal conditions in a production data center, it developed an acquisition protocol to monitor temperature and humidity from 52 homemade sensor nodes. The different environments monitored by RACNet and ThermoNet lead to discrepancies in WSN topology, data acquisition protocol design, and data analysis.

In terms of thermal instrumentation, traditional temperature sensors require wiring for the purpose of energy supply as well as information retrieval. One recent commercial alternative, THUM [7], provides a USB temperature/humidity sensor ($160) which connects to a PC through an available USB port (which adds to the cost). A representative wireless thermostat systems, by Venstar [9], costs $290 which replaces the existing room thermostat and has to be wired akin to a standard 24VAC
thermostat. Systems like the Venstar suit home environments but not the large-scale Dreese like HVAC systems. In contrast, ThermoNet is developed for large building environments, offering non-intrusive low-cost thermal instrumentation for the HVAC system.

In terms of improving HVAC efficiency, leveraging occupancy information to optimize HVAC schedules or adapt setbacks has received a lot of attention in recent research. Authors from UC Merced [29] showed using EnergyPlus simulations that it was possible to achieve 42% annual energy savings while still maintaining ASHRAE comfort standards by applying their occupancy-based ventilation control. These efforts considered a per-zone HVAC system in simulation tools, whereas, the main focus of our work is to evaluate the overall performance of a building wide HVAC system. In [11], Agarwal et al used occupancy information from a presence sensor platform to implement an occupancy-aware HVAC control scheme that obtained approximately 8% to 15% savings in energy use while controlling a floor of their four floor building. Their target building is six years old as of this writing and is controlled through a BACnet network that provided thermal measurements to researchers as well as access to each thermal zone. By comparison, Dreese is a relatively old building constructed in 1994 when BACnet was not the ASHRAE/ANSI standard yet. Nevertheless, Dreese represents the situation of a majority of buildings in our campus; hence, studying the effectiveness and efficiency of its HVAC system is materially useful.

5.3 Sensor Network System Design

Our design of ThermoNet is based on several key objectives. First, the system should be incrementally deployable without requiring costly modifications to
the building. Second, the system should be able to communicate reliably to every corner of the building for a period that lasts over years. In this section, we discuss the system architecture including costs of deployment, and the dynamic power adaptation component which extends the system lifetime to more than 2 years.

5.3.1 System Architecture

ThermoNet is a three-tier sensor network, consisting of a server and a backbone of devices that each supports tens of wireless sensor nodes. As shown in Fig. 5.1 (b), Tier 1 consists of the base station, a Dell PowerEdge series server, which collects and stores sensor measurements in a database. It also maintains and supports visualization of temperature and light statistics at different time scales.

Tier 2 consists of gateway nodes which are located one each in a control room on each floor, and 4-5 helper motes per gateway node. This backbone enables messages from sensor nodes to be forwarded to the server, and vice versa. These gateway nodes are Stargate 7.2 devices [5], which are 32-bit class Linux devices; and the helper motes are low-power TelosB motes [6]. Daemons running on Stargate forward packets received from helper motes to the base station and vice versa. In this deployment, a Stargate connects to the base station via department’s intranet. Stargates are wire-connected to their helper motes through USB adapters; the wires are strung through the false ceiling of a floor, which are hidden from building occupants, with helper mote locations being chosen carefully to ensure full coverage of the floor. Field experiments suggest that one floor of helper motes can communicate reliably with sensor nodes on 3 to 5 floors. Thus, a recommended backbone scale for Dreese requires three gateway nodes, which may be deployed on floor 2, 5, and 7. By constructing the backbone
sensing fabric, Tier 3 sensors are able to send and receive packets from almost every corner of the building, enabling a variety of applications running on top of it.

Tier 3 consists of TelosB temperature and light sensor nodes, deployed in almost every room of the building (printer, control, restricted rooms, and restrooms were excluded). These nodes report the measurements made in each room periodically in wireless fashion via the Tier 2 helper nodes. These nodes are powered by a pair of AA batteries and packaged in an enclosure we designed and had manufactured (see Fig. 5.2 (a)), so as to be aesthetically and location compatible with the control unit in each room. The cost per enclosure was about US$4; it includes small vents near the temperature sensor and has a hole close to the light sensor, to allow measurements. We typically placed the packaged sensor mote abutting (and lightly glued) to a room control unit; to the top of a white board or a bookshelf when the room control unit does not exist. The light sensor is welded with the TelosB mote and the temperature sensor is a TMP36 [8], which is low cost and is low-power as it consumes less than 50\(\mu\)A current. It provides a voltage output that is linearly proportional to the Celsius temperature. Before deployment, we tested the accuracy of TMP36 sensors by comparing their readings with that of a high-precision thermostat. We found that the sensor accuracy is within \(\pm 2^\circ F\) over the \(32 - 122^\circ F\) temperature range.

This partly-wireless, hierarchical sensing structure was chosen for multiple reasons. First, backbone nodes enable deployment for a broad range of sensing applications (other than temperature/light monitoring). Second, the tiered topology allows potential shifting of energy and computation intensive tasks from battery-powered sensors to wall-powered backbone nodes. It also simplifies maintenance to some extent since the backbone nodes are accessible in common (ceiling) areas, whereas the sensors are
in rooms where we need permission from occupants to enter. Third, the architecture enables debugging and reprogramming of TelosB motes from the Stargate.

In terms of cost, the gateway package on each floor consists of one Stargate ($300), 5 ethernet adapters ($30 each), 5 TelosB motes ($70 each in 2005), and one USB hub ($10), costing $810 per floor. The per-floor wiring of helper nodes to gateway nodes needed roughly 2 hours of labor, at a cost of $60. Each room requires one TelosB mote ($70 each) mounted with a temperature sensor ($0.61 each), a pair of AA batteries ($1), and an enclosure ($4), leading to a cost of $76 per room. A typical base station costs $500. Hence, the amortized per-room cost of ThermoNet with three floors of gateway nodes for monitoring 100 rooms in Dreese amounts to

$$C = \frac{500 + 810 \times 3 + 60 \times 3 + 76 \times 100}{100} = 107.1,$$

where the costs of deploying sensor infrastructure is amortized to approximately $37 per room. Note that the TelosB platform was designed for general WSN experiments; if a customized sensor mote were used, the per room cost would be reduced significantly.
5.3.2 Dynamic Power Adaptation

Except for the backbone TelosB nodes which are powered through USB adapters, about 100 Tier 3 TelosB motes rely on AA batteries for energy supply. Their two-year lifetime is achieved by duty cycling as well as power adaptation, as follows.

First, a Tier 3 node is duty cycled at approximately 0.003%, i.e., every 15 minutes the node turns on its radio to report data, waits for an acknowledgement from the backbone node for 20 milliseconds, and then switches off its radio. The base station keeps track of periodically-sampled node health information as well as instruments, containing voltage, transmission power level, and other network relevant information. Since backbone nodes are always on, packets sent from Tier 3 will be relayed to the server with trivial delay. Fig. 5.3 (a) shows the average voltage change with standard deviation on Tier 3 nodes over 12 months. The TelosB datasheet [6] suggests that the minimal supply voltage for mote operations is $2.1\text{V}$, thus, the expected lifetime of a typical Tier 3 node is more than two years, longer than the theoretical expectation given that a conservative power level (the highest) was used in our calculation. As will be described in next paragraph, the power adaptation scheme further reduces energy consumption on wireless nodes. When the supply voltage drops to the lower bound, AA batteries need to be replaced at Tier 3 nodes.

Second, the power adaptation component minimizes a node’s transmit power level subject to the constraint of achieving reliable communication. Since it is known that RF channels in buildings vary over time and space, in part due to the existence of a number of WiFi access points, it is thus infeasible for a node to maximize both reliability and efficiency by choosing a constant power level. Our solution for selecting power levels is illustrated in Fig. 5.2 (b). Initially each node starts with the highest
transmission power level (31). The power adaptation component periodically reduces the current power of a node by four levels (state=down) until the measured Packet Reception Ratio (PRR) drops below the threshold of 98%. Then, it increases power gradually to the lowest level that meets the PRR requirement (state=up). In addition to conserving energy usage, another advantage of this scheme is that Tier 3 nodes can lock to the backbone node(s) with best link quality in terms of communication cost and eliminate less reliable links. Data traces from ThermoNet reveal that a wide range of power levels have been adopted by Tier 3 in communication. We plot the probability mass function for transmission power levels used by Tier 3 nodes over a year in Fig. 5.3 (b), where the power range of 3 to 31 is divided into eight operating levels. It is shown that the dominant transmission power level has been reduced to the second lowest level (7) across the network.
5.4 Thermal Comfort Assessment

This section quantifies the level of (non-)compliance with the current thermal comfort objective, which lets us identify opportunities for improving the HVAC system as well as classify anomalies over various time durations. Specifically, analysis of season long indoor temperatures reveal seasonal mode change anomalies in the HVAC system; and based on the cumulative data, we identify *ill-conditioned rooms* whose proper conditioning require global/local controller adjustment or manual maintenance; in addition, we establish a basis for online detection of anomalous rooms on a daily basis.

5.4.1 Building Comfort Level

We begin by characterizing daily ratios of comfortable rooms to the total number of rooms monitored over a year. Given the control objective of $[70°F, 74°F]$, Fig. 5.4 (a) discriminates ratios of three types of rooms in terms of their daily average temperatures: hot ($> 74°F$), comfortable ($[70°F, 74°F]$), and cold ($< 70°F$) rooms. Note that the discontinuities of data points in the figure are due to occasional network maintenance performed on the Stargate devices. We observe that the ratio of the comfortable area to the whole area is on average 47% of the building: that is, more than half of the space is not well conditioned. This ratio is even lower when we apply the Green Building LEED Silver Certification policy, which mandates maintaining temperatures during heating and cooling periods at $70°F$ and $76°F$ with $±2$ degree error respectively when occupied [3]. *These measures provide strong evidence for refining the global control policy of the building; approximately 40% and 30% of*
we will discuss reduction of over-conditioning in Section 5.5.

5.4.2 Seasonal Mode Change Anomaly

Next, we characterize indoor daily temperature with respect to outdoor ambient conditions, such as air temperature and wind speed. Based on the year-long sensing data from ThermoNet and external weather conditions measured by an on-campus weather station [4], we compare the average indoor temperature, average outdoor air temperature, and maximum wind speed in Fig. 5.4 (b). The indoor temperature is defined as the mean of room daily temperatures, the value of which varies in the range of $[69^\circ F, 76^\circ F]$.

We find that the correlation coefficients between indoor and outdoor temperatures (indoor temperature and wind speed, respectively) over the year are within ±0.3, implying that ambient conditions have little influence on the building. More importantly, as indicated by the ovals in Fig. 5.4 (b), we discover that indoor temperatures tend to deviate from the norm during certain periods of time, typically when
season changes occur. For example, during the period of the second oval, the building was first overcooled and then overheated as outdoor air temperature started dropping in October. Our discussions with building automation colleagues suggest that this is likely due to a HVAC system malfunction, although there is some chance that if building policies were refined that these could be avoided. Either way, the use of the on-line information serves as feedback for building automation/maintenance so that this sort of anomaly can be avoided or handled more promptly.

5.4.3 Long-Term Ill-Conditioning

Analysis of year-long individual room data reveals that there are some rooms for which adjusting the global control parameters may not suffice to satisfy the thermal comfort objective. For instance, some rooms are always hot or cold no matter what the HVAC mode of operation is.

The percentage of time that a room’s daily temperature falls outside the acceptable range determines its level of ill-conditioning. To account for factors such as standard deviation of daily temperatures and accuracy errors in temperature sensor readings, we extended the comfort range by ±2 degrees. Thus, if the daily average temperature of a room is beyond the range of $[68^\circ F, 76^\circ F]$, the room is considered to be ill-conditioned on that day. Note that the daily temperature variation is not considered here since that 95% of daily standard deviation observations for all rooms over the year are within 2 degree error.

Fig. 5.5 (a) plots the cumulative distribution function for percentage of days over a year that a room has been ill-conditioned. We observe that approximately 80% of rooms are cumulatively over- or under-conditioned for less than 25% of the time (3
months in a year); a few rooms are ill-conditioned for most of the year. Accordingly, Fig. 5.5 (a) classifies rooms into three categories: persistently ill-conditioned, intermittently ill-conditioned, and (marginally) well-conditioned rooms, whose ill-conditioning ratios are respectively in a range of $[60\%, 100\%]$, $[25\%, 60\%]$, and $[0, 25\%)$. Incidentally, we also observe that the average length of contiguous ill-conditioned periods is highly correlated with the cumulative number of ill-conditioned days.

In other words, a majority of rooms have temperatures which over time oscillate within the borders of the acceptable range. We consider them to be well-conditioned or borderline rooms; the existence of borderline rooms is to be expected given the complexity of air distribution/diffusion in a large building; the number of these rooms can likely be controlled by proper selection of global control parameters.

The existence of a number of intermittently ill-conditioned rooms is an emergent finding of our analysis. That most of these rooms remain ill-conditioned for non-trivial lengths of time suggests that these rooms may need parameter adjustment at
the local controllers or even air distribution system maintenance. As for persistently ill-conditioned rooms, manual maintenance or modifications of the air distribution system may be required.

Fig. 5.5 (b) visualizes the result of our classification in a 2-D map of Dreese. The rows correspond to floor levels and rooms are displayed in a counter-clockwise with respect to the floor layout depicted in Fig. 5.1 (b). Due to differences in floor plan and room size, some floors have fewer rooms than others. We distinguish persistently hot, persistently cold, intermittently hot, intermittently cold rooms with red, blue, pink, and cyan colors respectively. Borderline and well-conditioned rooms are all colored in green since occasional thermal variations are difficult to avoid in the building. Fig. 5.5 (b) reveals that the number of cold rooms is larger than that of hot rooms.

We investigated a few ill-conditioned rooms and interviewed their occupants. Anecdotally, the staff and students working in persistently hot rooms complained to us that their room temperatures have been uncomfortable year round. Viz-a-viz persistently cold rooms on the 8th floor, room 886 serves as the data center for CSE department, which is independently maintained at a rather low temperature. An interesting observation is that a nontrivial number of ill-conditioned rooms are located at corners of the building, such as rooms 779, 679, 883, 281, 791, 691, 591, 399, 898, and 798, which suggests that local controllers around corners need careful adjustment to ensure their effectiveness. Some of the persistently or intermittently cold rooms are relatively large-space inner rooms, such as 480, 395, and 380 (376 is a small room next to 380). Since almost every individual room’s pneumatic controller can be adjusted locally (with a change in supply air temperature, supply static pressure, hot water
Figure 5.6: Anomalous rooms on Dec. 22, 2010, as identified by DBSCAN clustering algorithm (epsilon=0.15, minPoints=6). Rooms that reside outside the dotted oval are anomalies.

temperature, and etc.), the ill-conditioning classifier guides building operators to diagnose areas that demand local adjustment or maintenance or repair, either remotely or physically.

5.4.4 Daily Anomaly Detection

In addition to identifying long-term anomalies, another function of ThermoNet is to classify anomalous rooms on a daily basis using the high-fidelity data. We distinguish anomalous rooms from ill-conditioned rooms since the former involve short-term patterns that do not conform with the temperature profile of the majority of rooms, while the latter are determined by the static building control objective and the current AHU discharge set points. For instance, anomalous rooms include high temperature variation that may be caused by using personal space heaters or leaving a window open or extreme temperatures that may be caused by a fire.
In the anomaly detection algorithm, we characterize each room by a vector of two elements: the average temperature and the standard deviation over a day. For instance, the X and Y axes in Fig. 5.6 represent daily room temperature and standard deviation on December 22 in 2010, respectively. We then use a density-based spatial clustering algorithm, DBSCAN [31], to find outliers, which are classified as anomalous rooms in the building. In this case, two key parameters in the DBSCAN algorithm, i.e., the distance of neighborhood (epsilon) and the number of minimum points (minPoints), are set to 0.15 and 6, respectively. In Fig. 5.6, the rooms that reside outside the dotted oval are rooms that either have high temperature variation or extreme temperatures. For instance, room 779 which had low temperature and high variance on that day was probably due to leaks in its window(s). The other high variation room 585 was likely caused by the use of personal space heater since the room temperature was higher than that of a majority of rooms. Thus, on-line detection of anomalies provides evidence that helps building operators detect suspicious events (such as building leakage or using personal space heaters) and raise fire alarms.

5.5 HVAC System Efficiency Assessment

This section analyzes opportunities for increasing the building’s HVAC system efficiency in terms of energy consumption at different time scales as well as with occupancy awareness.

5.5.1 Reducing On-going Over-Conditioning

As seen in Fig. 5.4 (a), about 40% of the building is over-conditioned (blue) in the cooling periods from July to October, and 30% of the building is over-conditioned (red)
in the heating periods from November to May. This substantial over-conditioning of the building suggests that the control policy for setting supply air discharge temperature can be made less aggressive. An eQuest analysis of the building shows that 17% energy can be saved by adopting appropriate supply air temperature reset\textsuperscript{11}.

Fine-tuning the set point can be performed experimentally since ThermoNet on-line information feeds can be readily integrated with on-line control policy enforcement. In fact, the existing AHU set points are determined based on only three statically selected rooms for achieving control objective. Limited samples from the building tend to result in over/under-conditioning. Fine grain instrumentation from ThermoNet enables utilizing the average or an appropriate fraction of room thermostats to guide set points. For instance, the following message shows the content of the feedback from building to the controller, which includes time, the average temperature, standard deviation, the highest, and lowest temperature of the building, percentage of effective area, and etc.

\[(time, avgT, stdT, maxT, minT, ratio, ...).\]

5.5.2 Longer Temperature Setbacks

Fig. 5.7 (a) shows the diurnal thermal dynamics in Dreese. We compute for each room its hourly temperature averaged over a season and show the hourly indoor temperature as the mean of all rooms’ hourly values for each season. Consistent with previous observations, the temperatures in Fig. 5.7 (a) are actually warmer during the winter and colder during the summer due to over-conditioning. Furthermore, the

\textsuperscript{11}Even more energy would be saved by properly adjusting set points to conform to the Green Build policy.
hourly temperature exhibits two waves during a day with peaks occurring around 5am and 5pm, respectively. These waves delineate thermal dynamics in the building associated with HVAC cycles. Note that temperature variation between 12am and 7am is more substantial than that between 8am and 10pm as a result of the AHU shutdown during night-time. Nevertheless, the overall average temperature variation is within only 2 degrees. During the day, the variation is within 1 degree except for summertime when the influence of outside temperature on indoor conditions becomes slightly significant. Note that temperature changes slowly at night when the AHUs are shutdown, and by 1-1.5 degrees within an hour of the AHUs resuming.

Fig. 5.7 (a) suggests that the AHUs can sleep more without violating the building’s control objective. Towards estimating the increase of sleep period, we characterize approximately the occupancy of each room per hour by mining through illumination measurements acquired by ThermoNet. Basically, a room is assumed to be occupied if the light measurement is higher than a pre-determined threshold. ThermoNet sensors have been installed carefully to be away from any windows so as to minimize the
impact of outdoor illumination. However, the current scheme does not distinguish the case where occupants do not turn off the lights when they leave the room. In future work, adding motion detection sensors such as PIR and radar sensors to ThermoNet is planned to refine the room occupancy estimation. Fig. 5.7 (b) presents the percentage of occupied rooms during one quarter and during the break preceding it, which verifies that the building is largely unoccupied from 10pm to 7am. Correlating this information with air diffusion rates, we recommend setback periods to be 10pm to 7am over the quarter and 8pm to 8am over the break. This schedule would respectively save 3 and 6 hours out of the 18 working hours every day during the quarter and break, which would correspond to 16.7% and 33.3% energy savings.

5.6 Data-Driven Thermal Model

Towards optimizing the control strategy of the HVAC system, we take one step further to provide a more representative feedback to the HVAC system than a predefined set of three room temperatures. The feedback should be able to stand for the average temperature of all rooms in the building or the average deviation from the objective temperature (72°F), which are metrics that are used to adjust set points at the controller. To tackle this problem, we first divide rooms into several clusters that result in similar room temperatures under the centralized AHU handler. Second, a delegate room from each cluster is chosen to build a cluster-based thermal model for the building with high accuracy.
5.6.1 Cluster Identification

Each room is characterized by a vector of its average temperature in every 15 minutes during working hours of the HVAC system. Two rooms are considered “similar” if their absolute temperature difference are less than 2°F (i.e., 1.1°C) in 70% of temperature measurements over a year. The similarity matrix of rooms is transformed into clusters. Fig. 5.8 shows the clustering results, where five clusters are colored by pink, yellow, green, cyan and blue. Rooms that do not have data are left blank. One room 376 is colored gray, which is not grouped to any cluster because the room temperature is extreme low, around 64.5°F. Each cluster indicates a relatively “isolated” temperature zone in the building.

5.6.2 Cluster-Based Modeling

Given the five major clusters of rooms determined by year-long temperature measurements, we may compute the average temperature of the building, $\hat{T}$, or the average deviation with respect to the objective temperature, $T_{dev}$, in terms of an appropriate set of “head” rooms chosen from each of these clusters. For instance, the feedback $\hat{T}$
or \( T_{\text{dev}} \) may be represented by

\[
\hat{T} = \frac{\sum_{i \in \mathcal{V}} T_i}{|\mathcal{V}|} = f(T_{h1}, ..., T_{h5});
\]

\[
T_{\text{dev}} = \sqrt{\frac{\sum_{i \in \mathcal{V}} (T_i - T_{\text{ref}})^2}{|\mathcal{V}|}} = g(T_{h1}, ..., T_{h5}).
\] (5.1)

In other words, an aggregate temperature of 100 rooms in Dreese is represented by a function of five room temperatures.

To determine a proper set of rooms, we randomly choose one room from each cluster to build a set of cluster heads. Assuming the average temperature of the building has a linear relationship with that of the cluster heads, we need to identify parameters (\( \beta_i \) or \( \alpha_i \)) associated with the thermal model by some regression methods:

\[
\hat{T} = \sum_{i=1}^{5} \beta_i \cdot T_{hi}, \quad T_{\text{dev}} = \sum_{i=1}^{5} \alpha_i \cdot |T_{hi} - T_{\text{ref}}|.
\] (5.2)

Then, for each set of cluster heads, we perform a \( k \)-fold cross validation over year-long temperature measurements, where \( k = 4 \). Best candidates of cluster heads are selected according to the average root mean square error of their cross-validations.

<table>
<thead>
<tr>
<th>Object</th>
<th>Thermal Model</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Temperature</td>
<td>( T = 0.185 \cdot T_{295} + 0.484 \cdot T_{480} + 0.334 \cdot T_{489} + 0.111 \cdot T_{574} - 0.060 \cdot T_{591} )</td>
<td>0.25° F</td>
</tr>
<tr>
<td>Avg. Error</td>
<td>( T_{\text{dev}} = 0.078 \cdot T_{295} + 0.318 \cdot T_{480} + 0.192 \cdot T_{489} + 0.103 \cdot T_{574} + 0.143 \cdot T_{591} )</td>
<td>0.20° F</td>
</tr>
</tbody>
</table>

Table 5.1: Thermal model of selected cluster heads and their root mean square errors of 4-fold cross validation.
Figure 5.9: Picture of (a) experimental (solid) and simulated (dashed) average building temperature ($\hat{T}$), (b) experimental (solid) and simulated (dashed) average temperature deviation of all rooms from 72°F ($T_{dev}$).

The results are summarized in Table 5.1. Furthermore, Fig. 5.9 compares the predicted temperatures and measured temperatures over the testing dataset using the two models listed in Table 5.1.

5.6.3 Correlation of Responses of Cluster Heads

We further evaluate temperature change of these cluster heads at the granularity of half an hour as well as the standard deviation of these temperature changes among cluster heads at each half an hour. Fig. 5.10 shows the cumulative distribution function (CDF) of the standard deviation, which quantifies the similarity of responses among clusters under the same control inputs. We observe that in 99% cases the value of standard deviation is less than 1°F, indicating that clusters of rooms respond similarly to control inputs.
5.7 Conclusions

Our long-term data-driven analysis based on ThermoNet reveals a significant scope for improving the thermal comfort of the building. Not only is there a need to refine its control policy, fine-grain local adjustments and maintenance operations are needed associated with a significant number of rooms. The study also reveals a significant scope for improving energy efficiency. These suggest that there may be significant value associated with proceeding to couple the on-line information received from a long-lived WSN with the control system operation and facilities maintenance operation. Future work about ThermoNet will be discussed in Chapter 7.
Coordinating the duty cycles of nodes in low power wireless networks raises challenging stabilization issues. In this chapter, we show how to maintain duty-cycle coordination across the partitions of a static network of nodes. The idea is to synchronize the wakeup times of the nodes based on information carried by mobile “token” nodes between the otherwise disconnected partitions; the stabilization challenge is to deal with the corruption of state in both the static nodes and the mobile tokens. Our basic protocol assumes zero or more token nodes traversing disconnected static nodes in a circular order without overtaking each other. Refinements of our protocol accommodate richer patterns of token traversal and speeds.

6.1 Introduction

Energy-efficient operation is a basic requirement for battery powered wireless sensor networking. Almost-always-off operation of sensor nodes is thus the norm in any wireless sensor network application whose lifetime has to be nontrivial. In synchronous architectures, almost-always-off operation is achieved via ultra-low duty cycle processing. (By way of contrast, in asynchronous architectures, it is achieved
1% to 0.1% duty cycling is characteristic of the state-of-the-art in mote-scale wireless sensor networks.

Duty cycling entails two inter-related issues: one is the maintenance of coordination between the nodes so that they are repeatedly simultaneously up in order to communicate with each other when need be; and the other is choosing the appropriate rate of cycling so that nodes are neither contending excessively for the communication medium nor are they waking up excessively and performing wasteful idle listening. In some cases, coordination may be needed globally across the network while in others it may be needed only with respect to node neighborhoods. Analogously, the same duty cycle may be chosen at all nodes in the network or, should the load across the nodes vary, different rates may be chosen in each locality.

Stabilization of both the coordination and the rate selection aspects of duty cycling is an unavoidable consideration, not only for initializing wireless sensor networks, but also for dealing with faults, clock drift, configuration or environment change, traffic or application mode changes, etc. For the case of static, connected networks, the stabilization problem has received significant attention; see for instance the work on stabilizing time synchronization in wireless sensor networks [37]. In this chapter, we focus on the less studied but no less important case of mobile or partially mobile networks. We motivate the problem at hand via a case study.

6.1.1 Duty Cycling for Elevator Sensing

PeopleNet is a wireless sensor network currently deployed in Dreese Labs, our department building on the Ohio State campus. As the name suggests, PeopleNet
is about people-centric sensing; it comprises scenarios that involve sensing for the people, by the people, and of the people.

A realized PeopleNet scenario involves communicating the location of all the elevators in the building to a server, which in turn makes this information available in real-time via the web or, more usefully for the residents of the building, via a multi-hop low-power wireless network involving mobile hand-held devices.

As shown in Fig. 6.1, each elevator car has a battery powered mote sensor embedded in its false ceiling which sends a radio beacon every time the elevator door opens at any floor. Beacons from an elevator mote are heard by a static battery-powered “relay” mote which is mounted nearby the elevator egress at the corresponding floor. In turn, the relay communicates the beacon information to a wired infrastructure network that respectively forwards the beacons to a base station server. The base station in turn makes the elevator location available to the local cellphone network upon demand as well as to the web continuously, see Fig. 6.2.
Figure 6.2: A snapshot of PeopleNet elevator localization webpage.

Note that this scenario involves both mobile sensor nodes (on each elevator) and static relay nodes (on each floor). Duty cycling involves having both mobile and static nodes simultaneously up. When elevators do not move at all, i.e., at night, the rate of duty cycling should decrease to suit energy-efficient operation; conversely, when elevators use increases, the rate of duty cycling should increase.

In the particular case of our building, a static relay node cannot communicate directly with relay nodes on neighboring floors. The static relay nodes thus comprise a network of singleton node partitions. Achieving stabilization of almost-always-off operation within the wireless network itself—i.e., without resorting to using the out-of-band wired infrastructure network—thus needs the elevator to serve as a token carrying the coordination information between the floor relays. When elevators do not move at all, for instance at night, the network is disconnected and its partitions can gradually fall out of sync. When the elevators resume functioning, stabilization is needed to regain the duty cycle coordination across the relay and elevator motes.

6.1.2 Summary of the Results

We abstract the problem of energy-efficient duty cycling as achieving stabilizing wakeup coordination across a network of partitioned clusters of static nodes via mobile
token nodes that move between the clusters. To begin with, we further abstract each cluster as a single static node and consider the special case where the tokens traverse the static nodes in a unidirectional ring fashion without overtaking each other.

For this “unidirectional ring” model, where the static nodes cannot communicate with other static nodes but only via the mobile tokens, we design a stabilizing protocol by which all nodes converge to being simultaneously up according to a pre-selected global duty cycle.

We also present refinements of the basic protocol whereby: (1) tokens may not individually traverse the entire ring, but the set of all tokens cumulatively traverse the ring; (2) the ring topology is generalized to an arbitrary graph consisting of zero or more pre-specified unidirectional rings, each of which has the group token traversal criterion; and (3) the duty cycle is adapted depending upon the token traffic rates.

6.1.3 Organization of this Chapter

Section 6.2 discusses related work. Section 6.3 formulates the system model and the basic ring duty-cycling problem. Section 6.4 presents our stabilizing program for the basic ring, and its proof of correctness. Section 6.5 discusses extensions of the program to accommodate a richer token mobility model, underlying graph model, and traffic adaptivity. Finally, concluding remarks are in Section 6.6.

6.2 Related Work

Power management in wireless networks has been deeply studied in recent years, particularly for the radio because radio communication is a dominant power consumer among all components at sensor node. Power management is, broadly speaking, coarse-grain or fine-grain. Coarse grain duty cycling is exploited in [33]. By way of
contrast, fine grain duty cycling focuses on scheduling sleep and wakeup time for each node, and is the focus of this chapter.

**MAC.** Duty cycling impacts the MAC layer, in that coordination between potential communicators is necessary. MACs are, broadly speaking, synchronous or asynchronous. A number of synchronous MACs have been proposed for static networks, such as [70] and [63], but few have dealt with the stabilization issues associated with loss of coordination or mobility. O-MAC in [21] is an exemplar which has: It is a receiver-centric MAC protocol which implements a locally exclusive receiver wakeup schedule.

The setting in this chapter is different: there is an asymmetry between tokens and nodes which does not exist in O-MAC, as a result of which only the tokens need to discover the node; thus, neither the optimal unidirectional or bidirectional schedules of O-MAC apply in this setting. Asynchronous MAC approaches are considered more appropriate for mobile sensor networks for the reason that the neighborhood may change periodically [58]. However, continuously sending out long preamble for each send is a very energy inefficient way of avoiding neighborhood discovery.

**Stabilizing coordination and mobility.** The literature on stabilization has considered synchronization and mobility in a number of ways, of which we recall a few. In [37], a stabilizing converge-to-max protocol is presented that deals with clock skew of nodes and uses on bounded size variables, but does not explicitly consider stabilization of duty cycling. Earlier work of Herman achieved time synchronization by disseminating the leader value across a stabilizing spanning tree. [65] provides a constant time clock synchronization algorithm for synchronous and partially synchronous systems for the special case of initial synchronization. Under a virtual ring
model similar to ours, [23] presents a self-stabilizing solution to a mobile philosophers problem. [26] studies location management and routing in mobile ad-hoc networks using external “leader” information to achieve stabilization. [25] maintains a stabilizing structure rooted at a “leader” token whose mobility is controlled for exfiltrating data efficiently.

*Duty-cycle adaptivity.* Adaptivity of duty-cycles in the presence of traffic changes is important, since if the duty cycle is lower than required, collisions or sender buffer overflow result; but, if the duty cycle is higher than required, energy is wasted on idle listening. [20] discusses a local technique for stabilizing adaptation of the duty cycle to optimize energy-efficiency.

### 6.3 System Model and Problem Statement

The system consists of up to \(N\) static nodes and 0 or more mobile tokens. Each static node and token has a unique integer identifier (Id); we assume that token identifiers are smaller than node identifiers. Note that our use of tokens is in contrast to the standard notion of tokens in the stabilization literature, in which tokens refer to state at nodes (or state relationships between nodes); tokens here refer to independent computing entities.

For convenience, we henceforth refer to a static node as simply a node. Nodes are all isolated, in the sense that their communications cannot be heard by other nodes. In contrast, tokens are never isolated: at each point in time, each token can communicate with some one static node and, vice versa, that node can always communicate with that token. Communications are half-duplex, so at a given time a token can communicate with a node or vice versa, but both cannot simultaneously.
Note that multiple tokens may be in the vicinity of a given static node at any time. It is thus possible that tokens may hear each others’ messages.

To begin with, we assume that tokens move such that they visit all nodes according to a fixed unidirectional ring ordering of the nodes; we may thus regard the nodes as being organized in a virtual ring. We assume that there is a lower bound on the amount of time that a token may be in the vicinity of a node; informally speaking, this lower bound will imply that each token is able to exchange a synchronization message to and from with the node in question even when synchronization between the node and the token is lost. There is no upper bound on the time that a token visits with the node. So tokens may stop moving or move arbitrarily slow. However, tokens cannot overtake other tokens as they move around the ring. (We will generalize the mobility model in Section 4.)

A “slot” is a unit of time in which a node may send or receive a message, and/or perform some local computation. A “frame” is a contiguous sequence of some large number, \( m \), of slots. The time sequence at each node is divided into a sequence of frames. The ratio of the number of slots in which a node is awake in each frame to the number of slots in each frame is the duty cycle of the node. We assume the network is synchronous at the level of “slots”. (The assumption is readily removed, and is introduced only for ease of exposition.) If properly initialized, nodes and tokens are synchronous even at the level of frames.

*Fault Model.* Tokens may leave or join the system spontaneously. Nodes may leave or join the ring as well, subject to the upper bound of the node number not exceeding \( N \). The state of the tokens and the nodes may be arbitrarily corrupted. Nodes and tokens may become desynchronized at the level of frames. As a result of these faults,
the starting state for the protocol may be thus arbitrary. Also, the clocks of individual
nodes may exhibit skew during the operation of the system and this should ideally
be tolerated without much overhead.

_Problem Statement._ Required is a stabilizing protocol whereby the frames of each
token and node are synchronized so that visiting tokens and nodes can mutually
communicate at one or more well known moments during the frame.

### 6.4 Duty-Cycle Coordination in a Multi-Token Ring

Coarsely speaking, the central idea of our protocol is to let a leader token dictate
the frame schedule to all nodes and tokens in the system. This is programmed as
follows: When the leader token visits a node, the node directly synchronizes its frame
schedule to be consistent with that of the leader. In turn, when a non-leader token
visits a node which has more recently synchronized with the leader, the non-leader is
indirectly resynchronized with the leader via the more recent information at the node.
And so on: When a non-leader token that has been indirectly synchronized with the
leader visits with a node which has less recently synchronized with the leader, the
node may indirectly resynchronize itself with the leader. Thus, as long as tokens are
moving around the ring, all tokens and nodes become globally synchronized.

We refer to the information exchange between a token and a node to synchro-
nize the frame as a “synchronization-exchange”. In our protocol, synchronization-
exchange is programmed as follows. We let each node wake up in the first slot of
their frame. Each token sends its overall state in a message to the node which it is
visiting by waking up in the first slot of the frame. A node that receives a message
from the token sends its response in the next slot. Note that the token may not
receive a response, however, if the frames of the token-node pair are not synchronized or if a collision occurs with another token message during its original send.

Two basic issues now need to be addressed: (1) how to complete a synchronization-exchange between a token and a node when their frames are out of sync or when message collisions occur when multiple tokens are in the vicinity of a node; and (2) how to stabilize the synchronization information system-wide after faults happen.

The protocol deals with the first issue as follows. To regain communication should frames be out of sync, each node randomly beacons during one or more other slots in its frame. When a token does not receive its anticipated response, it remains awake until it hears the node beacon. Upon receiving the node beacon, the token can adjust the time for its next communication so as to ensure that the node is then awake and, by using random backoffs over the set of possible send times, to avoid collision with other visiting tokens. The next subsection explains how the protocol deals with the second issue.

6.4.1 Synchronization-Exchange Protocol Design

The protocol consists of four main components: leader election, continuous frame synchronization, false leader detection, and global reset, which are described next.

Leader Election. A leader token is elected so that its frame schedule serves as a global reference to synchronize the frames of all nodes and tokens in the system, which may initially be arbitrarily staggered. Stabilizing leader election is achieved as
follows. The up token with lowest Id is prioritized to win the election; a variable $l$ is
maintained at each node and token to store the lowest Id known.

*Continuous Frame Synchronization.* Each node and token maintains a recency/staleness
time estimate of the number of frames since it last received synchronization informa-
tion about the leader frame schedule, either directly from the leader token or indirectly
from some other token/node.

This count is maintained using an integer variable $c$. The $c$ value is assumed to be
(implicitly) incremented at the end of every frame in which no new synchronization
information is received. A special case is that the $c$ value of a leader token is always
zero. When a node encounters a token with the same $l$, whichever has the smaller $c$
dominate their pairwise synchronization of the frame schedule, since the smaller $c$
represents a more recent and therefore more accurate schedule. In other words, when
a node and token agree on the leader Id, the one with the larger $c$ value will adjust
its frame schedule to conform to that of the one with the smaller $c$ value.

Since non-leader tokens and nodes propagate frame synchronization information,
they may introduce errors because of their local clock skew. Tokens and nodes there-
fore make adjustments to compensate for their skew. Fig. 6.3 illustrates the schedule
adjustment and skew calculation. Node $j$ measures the relative difference between its
clock time as well as $c$ and the corresponding values at $i$, namely $T_{Err}$ and $d_c$, respec-
tively. Since $i$ has a smaller $c$, $j$ adjusts its frame schedule. By way of compensation
for skew, it calculates the $T_{Err}/d_c$ to be the average clock shift compared with the
leader and incorporates $T_{Err}/d_c$ to its skew estimate.

*False-Leader Detection.* Tokens are responsible for detecting whether the current
leader is no longer up, i.e., has left the system. Each token maintains the total
number of nodes visited by the token that have its $\langle l, c \rangle$ value, in a variable $cc$. Thus, if a token propagates its $\langle l, c \rangle$ to a node or the node already has the same value, the token increments $cc$; likewise, if a node has smaller $\langle l, c \rangle$ value than a token, the token copies the value of the node and resets its $cc$ to 1.

If the current leader is not up, the lowest $\langle l, c \rangle$ value in the system is propagated to other nodes in the ring because none of them would have met the corresponding leader more recently. One or more tokens would thus eventually detect that their $cc$ value has increased to $N$, where $N$ is an upper bound on the number of nodes in the ring, thereby detecting that the leader is false. Should tokens and nodes become desynchronized before a false leader is detected, we let nodes temporarily follow the tokens’ schedule.

**Global Reset.** Once a false leader is detected, the detecting node launches a new round of leader election in the ring. This objective could be simply realized using an integer sequence number, which is incremented to launch the new round. Upon seeing the higher sequence number, other nodes would reset their status and then participate in electing a new leader. Since the previous lowest Id has left the ring, a new lowest Id will succeed in the competition.
We bound this sequence number in size by using instead a two-valued (green and red) state variable $s$ to reset the ring. “Green” represents a normal state. In an ideal initial state, tokens and nodes would both be green. When a token detects that leader $l$ is not up, it updates $s$ to red, which indicates that their $\langle l, c \rangle$ information is outdated. Note that when a token changes its $s$ to red, it must be the case that the entire ring has the same $\langle l, c \rangle$. Therefore, if a node meets a red token, since both have the same $\langle l, c \rangle$, the node can follow the token and change into red. Only when all nodes and tokens have changed to the red state will a token reset itself into green, and thereby trigger other red nodes to reset themselves. The $\langle l, c \rangle$ value after reset cannot equal a false leader’s $\langle l, c \rangle$, therefore, when a red node or token encounters a different $\langle l, c \rangle$, the red node has to reset itself. Thus, the entire ring will be reset to a green state again. A new round of leader election is then started, having cleaned that false leader from the system. The state transition is implemented by actions $P_2$ and $P_3$ in next subsection and proof of convergence is presented in Lemma 4 in Appendix.

6.4.2 Synchronization-Exchange Protocol Variables

Each node and token maintains variables $l, c, s$; each token additionally maintains variables $p$ and $cc$. The associated semantics are summarized below:

- $l$ is the lowest Id currently known to the token/node. Recall that token Ids are lower than node Ids, hence $l$ at any node need never be higher than its node Id. Upon reset, $l$ is set to the local Id.

- $c$ is a count of the number of frames in which a node has not received synchronization information from the leader token.
• $s$ is the state of a token/node: 1 denotes the green state, and 0 the red state where false leader has been detected. 1 is the ideal initial state for both nodes and tokens.

• $p$ is the Id of the last node that the token has finished a synchronization exchange with. Since a token may visit with a node indefinitely, $p$ is used to distinguish whether or not the token has reached a new node. Its ideal initial value is -1 which means that the token has not yet synchronized with any node.

• $cc$ is, when $s = 1$, the count of nodes that a token has synchronized with that have the same $\langle l, c \rangle$ value; when $s = 0$, $cc$ is the count of nodes that a token has synchronized with that have the same $\langle l, c, s \rangle$ value.

• $token.i$ is true iff $i$ is a token.

• $Up.i$ is true iff the token $i$ is currently up.

The statement $reset i$ is defined to restore the initial value for token/node $i$:

$$reset i \triangleq l.i, c.i, s.i := i, 0, 1; (\text{if } token.i \text{ then } p.i, cc.i := -1, 0)$$

### 6.4.3 Synchronization-Exchange Protocol Actions

In this subsection, we present the actions of our stabilizing program at token (respectively, node) $i$ by which $i$ synchronizes with a node (respectively, token).

$$P_1: \quad token.i \land (\langle l, c, s \rangle.i = \langle l, c, s \rangle.j \lor (\langle l, c \rangle.i = \langle l, c \rangle.j \land s.i > s.j))$$

$$\land l.i \neq i \land p.i \neq j \land cc.i < N$$

$$\rightarrow cc.i := cc.i + 1; p.i := j$$
\[ P_2: \quad (\langle l, c \rangle . i = \langle l, c \rangle . j \land s . j < s . i \land \text{token}.j) \lor (\langle l, c \rangle . j < \langle l, c \rangle . i \land s . i = s . j = 1) \]
\[ \rightarrow \quad \langle l, c, s \rangle . i := \langle l, c, s \rangle . j; \quad \text{if token}.i \quad \text{then} \quad \text{cc}.i := 0 \]

\[ P_3: \quad \text{token}.i \land \text{cc}.i = N \]
\[ \rightarrow \quad s . i := s . i + 2; \quad \text{cc}.i := 0; \quad p . i := -1; \quad \text{if} \quad s . i = 1 \quad \text{then} \quad \text{reset} \quad i \]

\[ P_4: \quad l . i > i \lor (l . i = i \land (c . i \neq 0 \lor s . i \neq 1 \lor \text{cc}.i \neq 0)) \lor \text{cc}.i > N \]
\[ \lor (\langle l, c \rangle . i \neq \langle l, c \rangle . j \land s . i = 0) \]
\[ \rightarrow \quad \text{reset} \quad i \]

6.4.4 Synchronization-Exchange Protocol Correctness

As is standard in proofs of correctness of stabilizing programs, we: (i) identify an invariant predicate, i.e., a predicate that is closed in the program and is such that all the computations of the program starting from any state where the predicate is true satisfy the specification of the program, and (ii) show convergence from arbitrary states to the invariant, i.e., upon starting from an arbitrary state, every computation of the program eventually reaches a state where the invariant predicate holds.

Invariant. An invariant of our program, \( S \), is a conjunction of the predicates \( S.1, S.2, S.3, S.4 \) and \( S.5 \), defined below. Let \( T \) denote the set of tokens, \( Z \) denote the set of nodes in the ring, \( T \cup Z = U \). Also, let \( d(i, j) \) be the clockwise distance between nodes \( i \) and \( j \), i.e., the number of nodes between node \( i \) and node \( j \) (we are assuming
here that tokens move in a clockwise order.)

\[ S.1 \quad (\forall i,j \in U :: l.i \leq i) \]

\[ S.2 \quad (\forall i \in U :: l.i = i \Rightarrow (c.i = 0 \land s.i = 1 \land (token.i \Rightarrow cc.i = 0))) \]

\[ S.3 \quad (\forall i \in T :: (s.i = 1 \land p.i \neq -1) \Rightarrow (\exists j \in Z :: (l,c).j \neq (l,c).i \Rightarrow d(p.i,j) \leq N - cc.i)) \]

\[ S.4 \quad (\forall i \in T :: (s.i = 1 \land cc.i = N) \Rightarrow (\forall j \in U : (l,c).i = (l,c).j)) \]

\[ S.5 \quad (\forall i,j \in U :: s.i = s.j = 0 \Rightarrow (l,c).i = (l,c).j = \min((l,c).k | k \in U)) \]

The structure of the proof is as follows (details of individual sub-proofs are in the Appendix).

**Lemma 3.** \( S \) is closed in the synchronization-exchange protocol.

The following progress proofs assume that the token do not all stop moving.

Let \( H.1 = (S.1 \land S.2 \land S.3) \).

**Lemma 4.** \( \text{true} \) converges to \( H.1 \) in the synchronization-exchange protocol.

**Lemma 5.** \( H.1 \) converges to \( S \) in the synchronization-exchange protocol.

**Theorem 4.** The synchronization-exchange protocol is stabilizing with respect to the predicate \( S \).

Let \( R \) denote the stable states upon starting from any state where the invariant \( S \) holds where the up token with least Id is known as the new leader to the entire system.

Let \( k = \min(i|i \in T \land Up.i) \).

\[ R = (\forall i \in U :: l.i = k \land s.i = 1 \land (token.i \Rightarrow cc.i < N)) \]

**Lemma 6.** \( S \) converges to \( R \) in the synchronization-exchange protocol.
6.4.5 Synchronization-Exchange Protocol Analysis

First, we note that the synchronization time is equal to the leader election time. When a leader is acknowledged by all tokens and nodes in the ring, global synchronization is achieved, after which the synchronization is maintained by the movement of the leader.

Leader election happens within a constant number of rounds of each token circulating around the ring, in the model where no overtaking is allowed. From the proof of Theorem 1, upon starting from an arbitrary state, every token has the correct $cc$ and $p$ values within one round of circulating around the ring. If there is no “maximum” false leader (as defined in the proof), the system converges to $R$ within the time taken by that lowest ID token to circulate around the ring. Otherwise, the maximum false leader will first be detected by a non-leader token within one round time. After the detectors second round of circulation, all tokens and nodes change state to “red”. Thus, after each token traverses around the ring and before at most 2 rounds each, global reset is completed, i.e., no false leader exists any more. Finally, the system converges to $R$ after the true leader circulates the ring. Thus, the system converges to a state in $R$ in 4 rounds of circulation, where “round” is defined as the time for the lowest token to traverse the ring.

6.5 Model Extensions and Protocol Refinements

In this section, we refine the basic protocol to accommodate three extensions to the system model.

Token mobility patterns. While mobility in many operational settings is often predictable, tokens may not traverse across all nodes (let alone in order and without
overtaking other tokens). In our elevator setting, for instance, elevator tokens go back and forth across contiguous—but not necessarily all—nodes. We therefore extend the model to assume that cumulatively some set of tokens repeatedly traverse the ring in a given direction, even though individual tokens may stop or reverse course. 

To accommodate this mobility pattern, in which the elected leader token may not itself visit each node in the ring, we refine the basic protocol so that in addition to tokens carrying information about how many nodes have copied their potentially false \((l, c)\) value (via the variable \(cc\)), nodes also play the role of relaying this information to other tokens that pass by. Specifically, we let each node maintain the \(cc\) variable and update it during its synchronization-exchanges. Subsequent tokens that visit the node may inherit and propagate this value to other nodes. Thus tokens and nodes can continue to cooperate to detect false leaders.

More general topologies. The virtual ring is readily generalized to any graph that is the superposition of multiple rings, assuming that each ring has a set of tokens that collectively traverse it repeatedly. Note that the token sets of abutting (and, more generally, connected) rings may share tokens. Thus the individual token traversal pattern may be arbitrary.

To accommodate this generalization, we refine the protocol so that each node maintains independent state for each of the rings it participates in. Its frame schedule is thus effectively the union of the frame schedules of the respective rings; in other words, its duty cycle is the sum of the duty cycles for the respective rings.

Global/local adaptivity of duty cycle to the rate of token arrival. For each ring, its leader token may take on the responsibility for choosing the frame length \(m\), and accordingly globally changing the frame length when it visits (or other tokens on
its behalf visit) all nodes. To achieve global adaptivity, we refine the protocol with memory of the previous and the new frame length at tokens/nodes.

A complementary approach is for each node to estimate/predict the local rate of token arrival, and to correspondingly adapt the number of times it wakes up in each frame to receive messages from tokens. This local approach relates to the issue (1) discussion in Section 3 regarding collisions during local synchronization-exchanges, and is useful when the arrival rate across the network is spatially variable. In particular, when the number of tokens visiting a node changes, the node should accordingly adapt the number of slots per frame that it wakes up in (in other words, adapt its local duty cycle).

To achieve local adaptivity, we refine the protocol so that instead of being up to receive a message from a token in the first slot of each frame, each node broadcasts a beacon in the first slot of the frame. The beacon advertises its wakeup schedule for the current frame. (The beacon may be randomly reiterated during the frame to deal with frame resynchronization between visiting tokens and the node.) Tokens can then choose which wakeup slots to contend in, thereby decreasing the probability of collision. Conceivably, the beacon may even indicate slot assignments for the tokens which the node knows are visiting it.

6.6 Conclusions

In this chapter, we addressed an abstract duty-cycling problem for low-power wireless networks in terms of stabilizing maintenance of coordinated awake slots across isolated static nodes. The result could be as substantial as increasing the battery life of the network from a couple of weeks to several years.
Mobile tokens provided the information flow for achieving the desired coordination in our solution. But since token mobility is largely independent of the system, achieving stabilization involved a nontrivial consideration of a potentially cyclic dependency between token information and node information. Our solution works largely for networks comprising one or more rings; the problem of stabilizing to a common duty cycle globally across an arbitrary graph that is connected by mobile tokens deserves further attention.

We also presented global and local methods for adapting the duty cycle to avoid collision and idle listening. Our solution of node discovery by keeping tokens continuously awake until they hear node beacons exploited the asymmetry between tokens and nodes in semi-mobile wireless networks, but is not necessarily optimal, and would be another topic for further consideration.

6.7 Appendix

6.7.1 Closure of the Program

Lemma 3: $S$ is closed in the synchronization-exchange protocol.

Proof. Predicates $S.1$ and $S.2$ are closed under the program trivially. If $S.3$ holds before $P_1$ is executed and $cc.i < N - 1$, the distance to possible lower $\langle l, c \rangle$ value should be decreased by 1 since $\langle l, c \rangle.i = \langle l, c \rangle.j$; if $cc.i = N - 1$ before $P_1$ is executed, $P_2$ and $P_4$ are closed under $S.3$ trivially. After $P_3$ is enabled, if $s.i$ changes to 0, $S.3$ holds obviously; if $s.i$ changes to 1 by reset, $p.i$ becomes -1, thereby $S.3$ still holds. $S.4$ holds for $P_1$, $P_2$ and $P_3$ because $\langle l, c \rangle$ doesn’t change after execution. $P_4$ holds for $S.4$ trivially. Similar to $S.4$, $S.5$ is closed under the protocol.  \qed
Lemma 4: true converges to $H.1$ in the synchronization-exchange protocol, where $H_1 = (S.1 \land S.2 \land S.3)$.

Proof. By fairness, state predicates $S.1$ and $S.2$ would be satisfied by executing action $P_4$ continuously, which are purely local stabilization. Variable $p$ at token may have an arbitrary value in state true, however, we argue that this value will become correct after the token moves forward and communicates with the next node, because $p$ will be updated to the Id of the next node.

The $cc$ value at each token could also be an arbitrary value between 0 and N. Nevertheless, it only remains in finite illegal states because later on the token would either change its $\langle l, c \rangle$ or $s$ so that $cc$ would be decreased to 0, also, becomes a correct value from this moment. In particular, if a token’s $\langle l, c \rangle$ is replaced by other node, $cc$ is set to 0; otherwise, if it encounters larger or the same $\langle l, c \rangle$ at a node, $cc$ keeps increasing by 1 each time. Since previous $cc$ is arbitrary, it would reach N quickly. However, whenever it equals N, it will be set to 0 according to $P_3$, which is a correct value with respect to the new state $s$ or $\langle l, c \rangle$.

$S.3$ is satisfied obviously if token resets. Otherwise, when the token’s value is replaced by other lower $\langle l, c \rangle$, $cc$ becomes 1 because it is the first node that has the value met by the token. If a token copies its own value to a node or the node already has the same $\langle l, c \rangle$, $cc$ keeps increasing by 1. It indicates that a relatively lowest value has been copied to number of $cc$ nodes. Therefore, if there exists a token who carries a lower leader information, it must stay somewhere outside the sequence of
nodes with same $\langle l, c \rangle$, which is equivalent to the predicate $S.3$. Hence, $S.3$ would be satisfied when $cc$ becomes correct.

Thus, token and node would eventually converge to state $H_1$, where the first 3 predicates are satisfied and variables $cc, p$ are correct with respect to $\langle l, c \rangle$ and $s$.  

Lemma 5: $H.1$ converges to $S$ in the synchronization-exchange protocol.

Proof. We define the state that false leader exists in the ring as $S_{FL} = (\exists i \in U : token.l.i \land \neg Up.l.i)$ and the set of false leaders is defined as $FL = \{l.i \mid i \in U : token.l.i \land \neg Up.l.i\}$.

1. $H_1 \land \neg S_{FL}$ converges to $S$.

Let $l_0 = \min(l.i \mid i \in U)$. Since token $l_0$ is up, the lowest value of $\langle l, c \rangle$ in the system is $\langle l_0, 0 \rangle$. No matter what type of nodes token $l_0$ meets, nodes will copy the leader’s $\langle l, c, s \rangle$. There might be some tokens or nodes previously in state 0, however, when $l_0$ visits them, nodes must change to state 1 by reset. Since true leader always provides fresher $c$ values to visited nodes, according to $S.3$, the clockwise distance between token and the true leader cannot be decreased to 0, indicating that all nodes have copied the same value. Therefore, no token in state 1 will increase its $cc$ to N. $S.4$ is satisfied. Since no token changes $s$ from 1 to 0, hence, $S.5$ is satisfied.

2. $H_1 \land S_{FL}$ converges to $S$.

We define the “maximum false leader” as the token who carries the false leader information with the highest priority. In particular, $\langle l_0, c_0, s = 1, cc = 0 \rangle$ is the maximum false leader, where $\langle l_0, c_0 \rangle = \min(\langle l, c \rangle.i \mid i \in U)$, $s = 1$ and $cc$ is the lowest value.
According to program actions, the maximum false leader will copy this value to all nodes it has visited unless they are the same. After each action performed, cc increases by 1. Thus, the maximum false leader is reduced by increasing its cc. Eventually, it will reach N and according to S.3, there would be no lower leader in the system, therefore, S.4 is satisfied when all nodes have copied this value. Subsequently, the maximum false leader will change its state to 0, and propagate state 0 to all nodes it has visited. During this procedure, the maximum false leader decreases itself from \(\langle l_0, c_0, s = 0, cc = 0 \rangle\) by increasing cc again. Eventually, it will reach N again because all nodes kept value \(\langle l_0, c_0, s = 1 \rangle\), therefore, they may only change to state 0 after seeing the maximum false leader in state 0. Before the maximum false leader resets (decreases) itself again, it has propagated state 0 to N nodes. Since tokens do not overtake each other, at the moment that maximum false leader encounters 2N nodes with same \(\langle l_0, c_0 \rangle\) (at the end of first round, change from 1 to 0; at the end of second round from 0 to 1), all other tokens must have met at least N nodes with the same value. Therefore, all nodes and tokens who remain in state 0 would have the same value where S.5 is satisfied.

\[ \square \]

**Theorem 4:** The synchronization-exchange protocol is stabilizing with respect to the predicate S.

*Proof.* true leads to \(H_1\), \(H_1\) leads to S, therefore true leads to S. \[ \square \]

**Lemma 6:** S converges to R in the synchronization-exchange protocol.
Proof. We define $R$ as a stable state where starting from a state at $S$ eventually an alive token with minimum Id is known as the new leader among the entire system, while cleaning previous false leader information if there is any. $R = (\forall i : k = \min(i | i \in T \land Up.i) : l.i = k \land s.i = 1 \land (token.i \Rightarrow cc.i < N))$

1. $S \land R$ is closed.

\{S \land R\} \{R\}: Assume that $cc.i$ equals N-1 and the guard of $P_1$ is enabled. We argue that this claim cannot be true under $S \land R$. Because when $cc.i = N - 1$ the same $\langle l, c \rangle$ has already been copied to N-1 nodes. According to S.3, the clockwise distance between last visited node $p.i$ and the live leader is within one node. If $\langle l, c \rangle$ at this node is the same again, where $P_1$ is enabled, there cannot be a live leader $k$ who has a lower $c$ (0). Hence, $\langle l, c \rangle$ value at the rest node should not be the same, and $P_1$’s guard will not be satisfied. Therefore, the claim above is false. If $cc.i$ is less than N-1, $R$ still holds after execution. Obviously, $R$ is held under $P_2$, $P_3$ and $P_4$.

2. $S$ leads to $R$.

There are two phases. In the first phase, neither token nor node is in state 0. Let $l_0 = \min(l.i | i \in U)$. A variant function is provided as follows.

$$F_1 = \left(\sum_{i \in U} |l.i - l_0|, \sum_{i \in U} |s.i|, \sum_{i \in T} |N - cc.i|\right)$$

The first element decreases until all nodes have acknowledged leader $l_0$. If $l_0$ equals $k$, then $R$ is reached when the first element of function $F_1$ reduces to 0. Otherwise, the third element will decrease till the second element becomes less than $|U|$, the second phase is reached where state 0 appears in the system.
In the second phase, all nodes in state 0 have the lowest \( (l, c) \) by S.5, therefore, we provide the variant function which decreases till \( R \) is reached. Let \( (l_0, c_0) \) indicate the lowest value in the system.

\[
F_2 = (|i\mid i \in U : l.i = l_0|, \sum_{i \in U} |s.i|, \sum_{i \in U} |l.i - k|, | \sum_{i \in T} |N - cc.i|)
\]

When false leader information \( (l_0, c_0, s = 0) \) is in propagation, the second and last element keeps decreasing. Because when some token has detected the false leader, it will decrease its \( s.i \) and also trigger nodes to decrease their \( s.i \). After all tokens and nodes turn to state 0, the first detector starts reset so that the first element of \( F_2 \) is reduced. When \( F \) becomes \( (0, |U|, 0, |T|) \), which means that \( k \) is considered as the new leader in the system, state \( R \) is reached again.
Chapter 7: CONCLUSIONS AND FUTURE WORK

Duty Cycling for Energy Efficiency

One of the formidable challenges that distinguish sensor networks from traditional wireless networks is the scarce energy supply provided at each sensor node. Given the state-of-the-art technologies in Micro Electro-Mechanical Systems (MEMS), embedded computing, and communication hardware, duty cycling radios such that they are turned on and off periodically for communication has been considered as an effective way of increasing the lifetime of sensor networks from a couple of days to years. Thus, a decade of research has been conducted in how to schedule the radio for communications, which results in hundreds of duty-cycled MACs as well as cross-layer protocols. Although these protocols enrich the choices for WSN users, we are lack of a systematic way to quantify and compare the proposed protocols in terms of achieved energy efficiencies and their associated tradeoffs.

In this dissertation, we presented theoretical bounds on achievable capacity when duty cycling and an analytical framework to systematically quantify energy efficiency of duty-cycled wireless sensor networks with emphasis at the MAC layer. We first studied in theory the achievable throughput capacity in duty-cycled wireless networks, which represents the performance of the optimal scheduler. To further investigate the
achieved energy efficiency by the state-of-the-art MAC protocols, we formulated an analytical framework that can be instantiated to characterize performance of MAC classes over a broad range of network and protocol configurations. By evaluating representative MAC protocols in this unified framework, we find that the receiver-centric synchronous class consistently achieves the best or close to the best performance for various metrics across much of the configuration space. The findings corroborate that receiver centricity and local synchrony are two critical factors that determine energy efficiency. We applied the receiver-centric synchronous scheduler in the frequency domain and developed Chameleon, which is a multi-channel MAC protocol that uses all channels in the spectrum to maximize energy efficiency.

**Future work.** We have examined energy efficiency in both time and frequency domains for wireless and sensor networks, but the following dimensions warrant further investigation.

- Although our performance analysis of duty-cycled MAC protocols focused on a local view of the network, these MAC models can be leveraged in analyzing performance of upper layer protocols for low-power sensor networks. One improvement is to extend the modeling framework to accommodate more advanced network configurations, such as a wider range of traffic scenarios and multi-hop settings.

In multi-hop scenarios, the choices of routes may significantly impact global energy efficiency of the network. Thus, analytically quantifying the impact of routing on energy efficiency in comparison with that of link layer parameters
would provide valuable insights for cross-layer protocol design. We also observe that energy efficiency either increases or remains constant as local traffic increases. From a cross-layer perspective, this implies that upper layers can improve efficiency by accommodating the growing traffic with higher duty cycle rather than (say) dividing the traffic over two routes. We plan on exploring these findings in the design and implementation of cross-layer protocols to optimize the global energy efficiency in future work.

- The energy efficiency analysis in this dissertation considers the primary power consumer of a sensor node in general cases, which is the radio. However, a few environmental surveillance applications involve intensive sensing as well as computing tasks \([64, 36]\), thus, other components on the sensor platform such as sensing and computation may also contribute substantially to the total energy cost in addition to power consumption incurred by wireless communications. In these cases, the definition of energy efficiency needs to include multiple energy consumers. The tradeoffs among communication, sensing, computation, and other modules have to be evaluated together for overall energy conservation.

- This study only investigates energy efficiency in homogeneous wireless networks, where the capabilities of sensor nodes in network are considered to be the same. Heterogeneous sensor nodes with different abilities in communication and computation have been used in applications to achieve a balance between performance and cost. A systematic study of energy efficiency in heterogeneous networks as well as a comparative analysis with respect to homogeneous networks
are in demand to help WSN developers choose an appropriate architecture and protocol that meets the requirements of their applications.

WSN Applications

Provided that killer applications are the driving force for the development of wireless sensor networks, we introduced two instances of indoor sensing applications in this dissertation, namely ThermoNet and wireless elevator locator. ThermoNet aims at assessing and improving the duty cycle of HVAC systems in large buildings such that comfort and energy efficiency may be maintained simultaneously. Wireless elevator locator uses low-power sensor networks to track and broadcast the locations of public elevators online and to mobile devices in the building. Both applications dealt with the problem on how to duty cycle the system to achieve multi-years-long lifetime while meeting throughput and latency requirements. Different from previous analysis on homogeneous networks, a tiered topology is used in these applications which allows potential shifting of energy and computation intensive tasks from battery-powered sensors to a small set of wall-powered backbone nodes. Thus, energy efficiency in heterogenous network architecture can be obtained in a different way from that in homogeneous networks.

Future work. Although both systems have successfully provided high-fidelity environmental information of the investigated building over a year, there are several methodological concerns that warrant further improvement.

• Multiple dimensions of physical information are necessary to better evaluate building thermal conditions. As discussed earlier, the PMV thermal comfort
standard involves physical information other than temperature. Deploying different types of sensors such as humidity and radiant temperature will enable us to estimate building PMV metric more accurately. Collecting feedback from room occupants is another important factor to be considered in the future. Furthermore, the accuracy of occupancy prediction can be improved by adding motion detection sensors, such as PIR or radar sensors, which could lead to a design of a more energy-efficient per-room control policy. The cost of these sensors and their integration will serve as a key constraint in this respect.

- Network-based sensor calibration deserves further exploration. Even though errors of temperature sensors were sample tested to be within 2 degrees, a small fraction of sensors may have larger skews. Additionally, the accuracy of sensor outputs decreases as the circuit voltage supplied by batteries drops below certain threshold. In order to tolerate faults introduced by anomalous sensors, an in-field calibration scheme is indicated, which would discover anomalies via observations from correlated sensors and adjust the skew dynamically.

- The resolution of thermal monitoring can be increased by deploying multiple sensors in a room. Currently, only one sensor is installed per room no matter what size the room is, which raises the speculation that the location of the sensor may be inappropriate since there may be air stratification problems that prevent it from making a proper measurement. However, the majority of the rooms in Dreese are relatively small office rooms, and since our sensors are typically co-located with the local room controller, we believe this is a risk only if the room controller was already located in a problematic location during
building design. For the minority of large rooms in Dreese, it is reasonable to implement more than one sensor to characterize air dynamics better.

- Apart from improving the reliability and variety of sensor devices installed in the building, one remaining problem is to model responses of the cluster heads corresponding to adjustment of set points, denoted by \( \delta \), at the controller. The main difficulty is that the adjustment at the controller was not measured in ThermoNet. Energy consumption of the HVAC system in terms of adjustment of set points \( \delta \) is unclear, either. Thus, energy consumption function \( J(\cdot) \) and response function \( \Gamma(\cdot) \) in Eq. (7.1) need to be determined by collecting data from both building and control sides. In future work, we plan to collaborate with building automation department to collect data at the controller side and develop control strategies based on these models.

\[
\min_{\delta(k)} \sum_{k=1}^{N} \left\{ \sum_{i \in V} p_1 \cdot (T_i(k) - T_{ref})^2 \right\} + \sum_{k=0}^{N-1} p_2 \cdot J(\delta(k)) \\
= \min_{\delta(k)} \sum_{k=1}^{N} \left\{ p_1 \cdot T_{dev}^2(k) \cdot |V| \right\} + \sum_{k=0}^{N-1} p_2 \cdot J(\delta(k)) \\
\text{s.t.:} \\
T_{hi}(k) = \Gamma(\delta(k-1), T_{hi}(k-1)), \quad i = 1, \ldots, 5. \\
T_{dev}(k) = \sum_{i=1}^{5} \alpha_i \cdot |T_{hi}(k) - T_{ref}|. 
\]

- We also plan to enhance the current sensor fabric to better support multiple long-lived sensing applications. As more and more WSN applications are being accommodated on the sensor infrastructure in Dreese building, backbone nodes in the system often need to serve as data relays for multiple applications. How to facilitate co-existence of a couple of applications in an efficient way has been
an emerging challenge for us. A straightforward solution is to unify link layer communications used by applications. That is, all applications that are deployed on the sensor fabric have to conform to the same MAC level message format as well as the communication scheme. This incurs an overhead when transforming existing applications that use different MAC schemes to the MAC recognized by the sensor fabric.

An alternative solution we have investigated is to design a virtual MAC (or a unified MAC) for the backbone nodes. The virtual MAC identifies communications from mobile or distributed sensor nodes and communicates with them according to their respective MAC schemes. By bridging these applications at the MAC layer, multiple MAC protocols are allowed to co-exist in the system. This level of integration is transparent to application developers, allowing applications to choose their own MACs. However, this attempt has not been completed yet mainly due to two issues. First, the conflict between synchronous and asynchronous MACs is difficult to resolve without compromise at one of the classes. Second, the design of state-of-the-art MACs has not been standardized, thus, these protocols vary in many dimensions. It is a formidable task for the virtual MAC to be flexible enough to accommodate differences in a broad spectrum of MACs.
Bibliography


