Spring 5-2012

Probabilistic QoS Analysis in Wireless Sensor Networks

Yunbo Wang
University of Nebraska-Lincoln, yunbowang@gmail.com

PROBABILISTIC QOS ANALYSIS IN WIRELESS SENSOR NETWORKS

by

Yunbo Wang

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Doctor of Philosophy

Major: Computer Science

Under the Supervision of
Professors Steve Goddard and Mehmet Can Vuran

Lincoln, Nebraska

April, 2012

PROBABILISTIC QOS ANALYSIS IN WIRELESS SENSOR NETWORKS

Yunbo Wang, Ph. D.

University of Nebraska, 2012

Advisers: Steve Goddard and Mehmet Can Vuran

Emerging applications of wireless sensor networks (WSNs) require real-time quality of service (QoS) guarantees to be provided by the network. Traditional analysis work only focuses on the first-order statistics, such as the mean and the variance of the QoS performance. However, due to unique characteristics of WSNs, a cross-layer probabilistic analysis of QoS performance is essential. In this dissertation, a comprehensive cross-layer probabilistic analysis framework is developed to investigate the probabilistic evaluation and optimization of QoS performance provided by WSNs. In this framework, the distributions of QoS performance metrics are derived, which are natural tools to discover the probabilities to achieve given QoS requirements. Compared to first-order statistics, the distribution of these metrics reveals the relationship between the performance of QoS-based operations and the probability to achieve the performance. Using a Discrete-Time Markov queueing model in node-level analysis and fluid models in network-level analysis, the distributions of end-to-end delay, the network lifetime, and the event detection delay are then analyzed. Based on the evaluation of QoS metrics, a probabilistic optimization framework is developed to demonstrate the investigation of the optimal network and protocol parameters. Guidelines of designing networks and choosing optimal parameters for WSNs are provided using the optimization framework. Intensive testbed experiments and simulations are used to validate the accuracy of the proposed evaluation and optimization framework.

DEDICATION

I wholeheartedly dedicate this dissertation to my girlfriend, Xijing Tan, for her love, support and sacrifices during the later part of my Ph.D. study. I would especially thank her for encouraging me in hard times, for standing with me all the time, for taking care of my daily life, and most importantly, for completely trusting me. I cannot imagine finishing this dissertation without her support.

I would also like to dedicate this dissertation to my parents – my mother, Ms. Shuping Peng, and my farther, Mr. Naiguang Wang – for their support and patience during the past six years. Being an engineer himself, my farther has helped me greatly in my study and research.

I would like to further dedicate this dissertation to all my friends. I especially thank Mr. Ying Tan for sharing the same favorite soft drink with me and for teaching me playing bridge. I would also thank Mr. Xin Dong for cooking three-flavored chicken for our dinner. Without their ideas of entertainment, I would have been rotten beside my computers.

ACKNOWLEDGMENTS

I would express my thanks and appreciation to my adviser, Professor Steve Goddard, for his guidance throughout my Ph.D. program. I especially thank him for inspiring me to solve problems like a researcher does, for constantly reading my poor-written drafts, and for tolerating and encouraging when I made mistakes. Without his help, I couldn't have even started my Ph.D. research.

I would also like to express my appreciation and admiration to my co-advisor and mentor, Professor Mehmet Can Vuran, for his tremendous help and direction. Being energetic, enthusiastic, wise, keen, persistent, knowledgable, and considerate, he has set a great example of how researchers should be. I could never gone this far without his guidance.

I am also indebted to my committee members, Professor Lisong Xu and Professor Lance C. Pérez, whose comments and suggestions on my research topics have helped me keep my dissertation in the right direction.

I would also thank my friends and lab mates, Mr. Xin Dong, Mr. Dave Anthony, and Mr. Paul Bennett. The conversations with these people have always been a great pleasure, and have sparkled a lot of great ideas which eventually became a part of this dissertation.

And last but not least, I would thank Holland Computing Center of the University of Nebraska and especially Professor David Swanson. The computer simulations and model evaluations in this dissertation were partially completed utilizing the FireFly supercomputer in Holland Computing Center.

GRANT INFORMATION

This dissertation is supported, in part, by grants from the National Science Foundation (0707975) and the Air Force Office of Scientific Research (FA9550-06-1-0375).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless sensor networks (WSNs) have been utilized in many applications as both a connectivity infrastructure and a distributed data generation network due to their ubiquitous and flexible nature [5]. Increasingly, a large number of WSN applications are investigated with various quality requirements for different network services specific to low-cost hardware, and unpredictable environment conditions [4, 15]. These requirements necessitate a comprehensive analysis of the Quality of Service (QoS) provided by the network. Based on this analysis, an optimization tool for network and protocol design is also essential.

QoS issues and techniques have been intensively investigated for ATM networks [19, 62], IP networks [7, 61, 62], and traditional wireless networks [14, 64]. In these studies, the evaluation of QoS is mainly focused on the communication quality characterized by communication delay, jitter, bandwidth, and loss rate. Traditional metrics, however, cannot fully characterize the QoS in WSNs [15], because of the distinct characteristics of WSN applications as listed in the following.

First, WSNs are utilized for a different set of applications from those with traditional networks [5]. These applications emphasize different characteristics of the

network and require different *services* to be provided by the network. Thus, the metrics to evaluate the *quality* of these services are also different from traditional QoS metrics. For example, for most WSN applications, sensor nodes are powered by batteries with limited capacity, and replacing the batteries is difficult. Thus, the network lifetime under battery constraints is a QoS measure that is more important than in traditional network analysis. Other examples of such QoS measures include the delay for event detection, and sensing rate of individual sensors.

Second, the environment conditions of WSNs are in nature unreliable and random. Sensor nodes are usually manufactured *en masse* with low-cost hardware. Many applications in harsh environments such as wild fields and battlegrounds further impose possible physical damage to the nodes [5]. Thus, it is expected that the nodes may randomly cease to work, resulting in a random network topology. Moreover, the wireless communication among nodes are also prone to random noises due to low-profile radio transceivers and limited communication power. All these random factors result in a large variance in QoS metrics, and cannot be thoroughly evaluated using traditional approaches, such as mean delay analysis [1, 9, 40], or worst-case analysis [12, 32].

Finally, due to limited resource availability, QoS analysis must be performed in a cross-layer manner. In traditional network analysis, with adequate resources assumed, the maintainability and modularity are emphasized at the expense of additionally consumed resources such as storage, computing power, and energy supply. Hence, the QoS is separately provided by different network layers. In contrast, with limited resources, WSNs are usually designed to exploit cross-layer operations and meet QoS requirements more efficiently [63]. For example, cross-layer integration can lead to significant energy conservation [93, 95]. Moreover, requirements on different QoS metrics can contradict with each other, and a tradeoff must be made to provide opti-

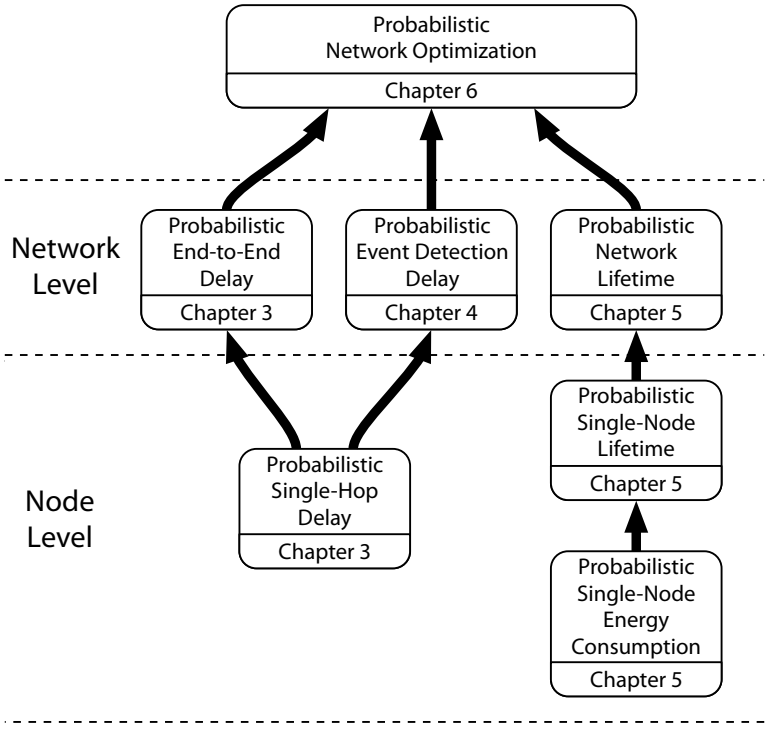


Figure 1.1: The structure of research objectives in this dissertation.

mal services. For example, lower delay and longer lifetime are usually contradicting design goals. Lower delay usually requires a high duty cycle, whereas longer lifetime usually favors low duty cycle. Therefore, a QoS analysis framework that captures the tradeoffs for the protocols and operations in the entire software stack is desirable.

In this dissertation, we provide a comprehensive cross-layer probabilistic analytical and optimization framework to evaluate the QoS provisioning in WSNs. The research objectives and solutions are discussed in the following.

1.1 Research Objectives and Solutions

As shown in Figure 1.1, the research objectives in this dissertation are structured in three levels. The bottom level and the middle level are the probabilistic analysis of

QoS metrics, whereas the top level is a probabilistic network optimization framework. At the bottom level, i.e., the node level, the single-hop delay, the single-node energy consumption, and the node-level lifetime are studied. At the middle level, i.e., the network level, the end-to-end delay, the network lifetime, and the event-detection delay are investigated. Based on the analysis of these QoS metrics, an optimization framework is then developed to aid network and protocol design. Due to the vast diversity in services requested by WSN applications, it is infeasible to provide QoS analysis in every aspect of application requirements. Therefore, we focus on the delay and the lifetime metrics, which are important for most applications.

In the following, the rationale behind the probabilistic investigation of these QoS metrics is explained in detail.

1.1.1 Probabilistic Packet Delay Analysis

One of the most important QoS metrics in WSNs is the packet communication delay. Characterizing communication delay in distributed systems has been investigated in different contexts. The latency performance of WSNs in terms of its first order statistics, i.e., the mean and the variance, has been analyzed in recent studies [1, 9, 40]. However, complex and cross-layer interactions in multi-hop WSNs require a complete stochastic characterization of the delay. Several efforts have been made to provide probabilistic *bounds* on delay. As an example, the concept of Network Calculus [20] has been extended to derive probabilistic bounds for delay through worst case analysis [12, 32]. However, because of the randomness in wireless communication and the low power nature of the communication links in WSNs, strict worst-case analysis cannot capture the stochastic behavior of end-to-end delay. Moreover, real-time queueing theory has been exploited to provide stochastic models for unreliable

networks [55, 101]. However, these models assume a heavy traffic rate, which is not applicable for WSNs.

Recently, probabilistic analysis of delay has been performed for broadcast networks [8, 70, 79, 80, 87] considering several medium access control (MAC) protocols. Indeed, the cumulative distribution function (*cdf*) of the delay for a given deadline can be used as a probabilistic metric for reliability and timeliness. However, while channel contention has been adequately modeled in these studies, additional delay due to multi-hop communication, queuing delay, and wireless channel errors have not been captured. Capturing these cross-layer effects is imperative to completely characterize the delay distribution in WSNs.

One of the goals of the proposed analytical framework is to provide comprehensive analysis for the delay in WSNs, among other QoS metrics. The delay distribution is an important metric to evaluate the communication services provided by the network, since it measures the probability that the network meets a given deadline. The developed framework highlights the *relationship between network parameters and the delay distribution* in multi-hop WSNs. Using this framework, real-time scheduling, deployment, admission control, and communication solutions can be developed to provide probabilistic QoS guarantees.

1.1.2 Probabilistic Event Detection Delay Analysis

Event monitoring is another important service provided by WSNs beside the packet communication. In typical event monitoring applications, numerous sensor nodes are deployed in the space, and operate collaboratively to monitor, report, and react to various physical events. When an event of interest occurs, it is detected by sensor nodes. Reports are then generated and forwarded to a sink via multi-hop commu-

nication. Based on the received reports, the sink may detect the event and perform appropriate actions, e.g., inform the forest administration in case of a fire. For such systems, reports must be delivered to the sink in a timely manner. Therefore, one of the most important performance metrics for event monitoring WSNs is the *event detection delay* [39], i.e., the delay between when an event occurs in the physical world and when a sufficient number of packets are delivered to the sink. Clearly, the event detection delay consists of two parts: the *discovery delay* for individual nodes to sense and detect the event, and the *delivery delay* for the network to relay reports to the sink. Analyzing the event detection delay is a crucial task for real-time WSN applications, which require predictable event detection delay guarantees to be provided by the network.

Traditional event detection delay analysis [12, 32] cannot capture the statistical characteristics of the event detection delay. Additionally, the event detection delay analysis is more complex than communication delay analysis in that, as event reports from individual nodes can be unreliable, it is more desirable to detect an event collectively from *multiple reports generated by multiple sensor nodes* [2, 39]. Thus, an event is generally considered to be detected only when a given number, n, of reports are received by the sink [38, 105].

To address these challenges, a probabilistic analytical framework is developed in this work to capture the delay characteristics of event detection in large-scale 2-D WSNs. A spatio-temporal fluid model is developed to derive the distribution of event detection delay. Accordingly, the mean event detection delay and soft-delay bounds for event detection can be modeled. The soft-delay bound (or p-delay bound) for delay is defined as the delay within which an event is detected with a given required probability p. Indeed, a lower p-delay bound indicates that the events can be reliably detected within a lower delay. Hence, the network is more desirable

for real-time applications. The empirical validations and simulation studies reveal that the developed model is suitable for high density networks and low traffic rate applications, common features of a large class of WSN applications.

Motivated by the fact that queue build up in low-rate traffic is negligible, a lower-complexity model is also developed. This model extends the delay analysis for single packets, and derives the event detection delay by first obtaining the end-to-end delay for each packet. This approach requires lower computational power than the first model by reducing the computational complexity from $O(A)$ to $O(\sqrt{A})$, where A is the area of the network.

1.1.3 Probabilistic Energy Consumption and Lifetime Analysis

In most Wireless Sensor Network (WSN) applications, nodes are powered by batteries, and replacing the batteries is usually a tedious work. When energy is depleted, nodes become inactive, losing their sensing and communication functionalities. Therefore, providing adequate lifetime is an important QoS in WSNs.

Accurately characterizing increasingly complex energy-saving techniques [5, 6] is challenging task. At the MAC layer, periodic sleeping based protocols [11, 78, 100] have been developed, where nodes are forced into sleeping mode periodically, while still maintaining network connectivity. At the network layer, energy-aware routing protocols [49, 83, 95] are also utilized to further reduce the energy consumption. Complicated network activities in multiple protocol layers necessitate a comprehensive and generic model to accurately evaluate the energy consumption in WSNs.

Traditionally, energy analyses are focused on the *average* power consumption. For example, in studies proposing the aforementioned energy-efficient WSN protocols

[11, 49, 78, 83, 95, 100], attempts to reduce the *average* energy consumption are made. Moreover, average energy consumption is the main focus in existing generic energy analysis models [42, 97]. Existing lifetime analysis studies, such as [48], also provide models for the *average lifetime*. However, due to the random nature of the wireless environment, in critical applications where a highly reliable network is required, only knowing the average energy consumption and average lifetime is insufficient.

Instead of *average* energy consumption and lifetime, their *probabilistic distributions* is investigated in this work. The distribution of energy consumption and lifetime captures the probability that the consumed energy within any given period is lower than a given value, and the probability that the lifetime is longer than a given period. This allows making trade-offs between the desired lifetime and the probability to achieve the lifetime. A Markov process-based model is developed to analyze the distributions of energy consumption and lifetime in WSNs. It is shown that when the given period is large enough, energy consumption *converges to a Normal distribution*. The analysis is validated by realistic testbed experiments and extensive simulations.

1.1.4 Probabilistic Network Optimization

Given the developed analytical framework, a natural question is how to exploit the analysis to aid network design. This question generally is equivalent to an optimization problem. In the evaluations of the probabilistic analysis, we use the framework to demonstrate the relationship between QoS performance metrics and network parameters. Indeed, requirements on different QoS metrics can conflict with each other, and tradeoffs must be made to provide optimal services. Therefore, an optimization framework that captures all QoS metrics is needed.

In this work, an important part of the developed analytical framework is a prob-

abilistic optimization framework that captures various QoS metrics. As shown in Figure 1.1, the framework is based on the analysis for key QoS metrics, such as the end-to-end delay and the network lifetime. The main purpose of the optimization framework is to demonstrate how to make decisions on choosing the optimal network parameters according to application requirements.

Beside the probabilistic metrics, We also investigate deterministic performance metrics that are not necessarily analyzed with a probabilistic approach in this dissertation, for example, the traffic throughput in monitoring applications. Moreover, we are interested in a set of network parameters as control variables, such as the network density, the traffic generation rate, and the duty cycle. We then formulate the probabilistic optimization problem as to find the minimum or maximum value for an objective metric, given a set of probabilistic or deterministic constraints on QoS performance metrics.

To solve the formulated probabilistic optimization problem, a heuristic technique that utilizes multiple local searches is developed. Using this technique in a case study, the optimal network parameters are investigated for a WSN with the anycast protocol. Extensive numerical results are obtained to validate the accuracy of this technique. From the optimization results, trends and insights about the tradeoffs in network designs are obtained.

1.2 Key Contributions

The key contributions of this dissertation are summarized as follows.

- **Formal Definition of Probabilistic QoS Metrics**

  In this work, we formulate formal definitions of probabilistic QoS performance metrics in WSNs. The important metrics discussed in this work are: the end-to-

end delay distribution, the event detection delay distribution, and the network lifetime distribution. As an intermediate step to investigate these metrics, two additional fundamental metrics are also defined, i.e., the single-hop delay distribution and the single-node energy consumption distribution.

- **Analytical Framework to Evaluate the Probabilistic QoS Metrics**
  A probabilistic analytical framework is proposed to evaluate the QoS performance metrics. Extensive testbed experiments and computer simulations are conducted to validate the accuracy of the framework.

- **Insight of How Network Parameters Affect the QoS Performance**
  Using the proposed analytical framework, an optimization framework is also proposed to derive the optimal network and protocol parameters. This framework can be exploited to aid the design and evaluation of network parameters and protocols before actually deploying the networks.

1.3 Dissertation Organization

This dissertation is organized as follows. In Chapter 2, the methodology of the probabilistic QoS analysis and optimization framework is discussed, including the system models, approaches, case studies, and validations. Then, each of the four major parts in this dissertation is presented in individual chapters. The probabilistic end-to-end delay analysis is provided in Chapter 3. The probabilistic analysis of event detection delay is described in Chapter 4. Then, the probabilistic energy consumption and lifetime analysis is developed in Chapter 5. In Chapter 6, the probabilistic network optimization framework is described. Finally, conclusions are given and open research problems are provided in Chapter 7.

Chapter 2

Methodology of the Probabilistic QoS Analysis

In this chapter, the methodology of this dissertation is described in the following aspects. First, the system models, including the network topology model, the wireless channel model, the traffic pattern models, are presented in Section 2.1. Then, the approaches and techniques used in the analysis are briefly discussed in Section 2.2. These include the node-level analysis using a discrete-time Markov process (DTMP), and network-level spatio-temporal fluid models. The third aspect is the case studies conducted to validate our framework. We present two different scenarios of case studies in Section 2.3 with different MAC and routing protocols: a TinyOS CSMA/CA MAC protocol combined with routing protocols with static routing paths, and an Anycast cross-layer protocol. These case studies are conducted using testbed experiments and simulations, for which the setup and techniques are described in Section 2.4.

2.1 System Models

In our analysis, we consider a network composed of sensor nodes that are distributed in a two-dimensional (2-D) field. Sensor nodes communicate with each other through a multi-hop route in the network. Each of the sensor nodes are capable of generating packets, which have specific destinations through multi-hop communications. Moreover, each node can also relay packets sent from other nodes to progress towards their destinations. Whenever a packet is generated or received for relay when a node is in the process of transmitting a packet, the new packet is temporally stored in a queue located in the node memory. Due to limited storage resources, the queue capacity is usually limited. Moreover, most nodes are powered by batteries with limited energy capacities.

2.1.1 Network Topology

The network topology in WSNs is determined by the way sensor nodes are deployed. We consider a network of N sensors that are distributed in a 2-D field. Two different types of network deployments are investigated.

- *Random deployment*: Individual sensor nodes are located randomly in a 2-D plane.
- *Deterministic deployment*: As a special case, we consider deployments, where sensor nodes are located at deterministic locations, e.g., grid topology.

In both cases, each node is identified according to its location $\boldsymbol{x} = (x, y)$ in Cartesian coordinate systems, or $\boldsymbol{x} = (r, \theta)$ in polar coordinate systems, and is characterized by its input traffic rate, $\lambda(\boldsymbol{x})$, queue length, $M(\boldsymbol{x})$, and battery capacity, $C(\boldsymbol{x})$.

2.1.2 Channel Model

Due to the channel noise and predominant shadowing/fading effects in the wireless channel, the communication between two nodes cannot be assumed to succeed all the time. Instead, there is a chance, which is measured by the *packet error rate (PER)*, that the packet transmission fails, whenever one or more bits in the packet are corrupted. In this work, a log-normal fading channel model is considered [107] to derive the PER, as summarized in the following.

In the model, the packet error rate depends on the transmission distance, transmission power as well as random multi-path fading and shadowing effects. Therefore, the packet error probability is a random variable (r.v.). In this work, we use the mean value of this r.v. to represent the packet error rate PER. The accuracy of this assumption is validated by our testbed experiment results throughout this dissertation. Accordingly, the r.v. received signal to noise ratio (SNR) $\Psi(d)$ at the receiver is a function of transmission distance d, and is given by

$$\Psi_{\mathrm{dB}}(d) = P_{\mathrm{t}} - P_{\mathrm{n}} - PL(d_0) - 10\eta \log_{10}\left(\frac{d}{d_0}\right) + X_{\sigma}, \tag{2.1}$$

where P_{t} is the transmit power, and P_{n} is the noise power in dBm. $PL(d_0)$ is the path loss at a reference distance d_0, η is the path loss exponent, and X_{σ} is the random shadow fading component, which is modeled by a zero-mean Normal distribution with standard deviation σ. Therefore the probability that the received signal SNR $\Psi_{\mathrm{dB}}(d)$ is higher than some value ψ is

$$\Pr(\Psi_{\mathrm{dB}}(d) > \psi) = \frac{1}{\sqrt{2\pi}} \int_{\psi}^{\infty} \exp\left(-\frac{\beta^2(d,\psi)}{2\sigma^2}\right) \mathrm{d}\psi = Q\left(\frac{\beta(d,\psi)}{\sigma}\right), \tag{2.2}$$

where

$$\beta(d,\psi) = \psi + P_{\mathrm{n}} - P_{\mathrm{t}} + PL(d_0) + 10\eta \log_{10}\left(\frac{d}{d_0}\right). \tag{2.3}$$

The expected bit error rate is a function of the expected received SNR, and then, the expected packet error rate is determined by the expected bit error rate. The derivation depends on the modulation and coding approaches the radio transceivers use. More detailed information about the derivations for typical transceivers are listed in [107].

In the analysis throughout this dissertation, this expected packet error rate, PER, is calculated, and is used to characterize the channel condition.

2.1.3 Traffic Pattern Models

In this dissertation, the traffic pattern is defined by the interarrival time distribution of packets, which is further determined by the application and protocols used. In a typical multi-hop WSN, the input traffic at each node consists of two parts: *locally generated packets* and *relay packets*. Locally generated packets consist of the local information sampled by the sensors, whereas relay packets are those received from the neighbors, and should be forwarded towards their final destinations. The inter-arrival time of the locally generated packets depends on the application requirements, with which the sensor data are generated, and the relay traffic depends on the network parameters.

To lay the foundation of the analysis in this dissertation, in the following, we aim to find the inter-arrival time of the locally generated packets and relay packets at each node for various applications. Specifically, two types of applications are considered: *event-based applications* and *monitoring applications*, depending on how the sensor

data is generated.

2.1.3.1 Locally Generated Traffic Pattern: Event-Based Applications

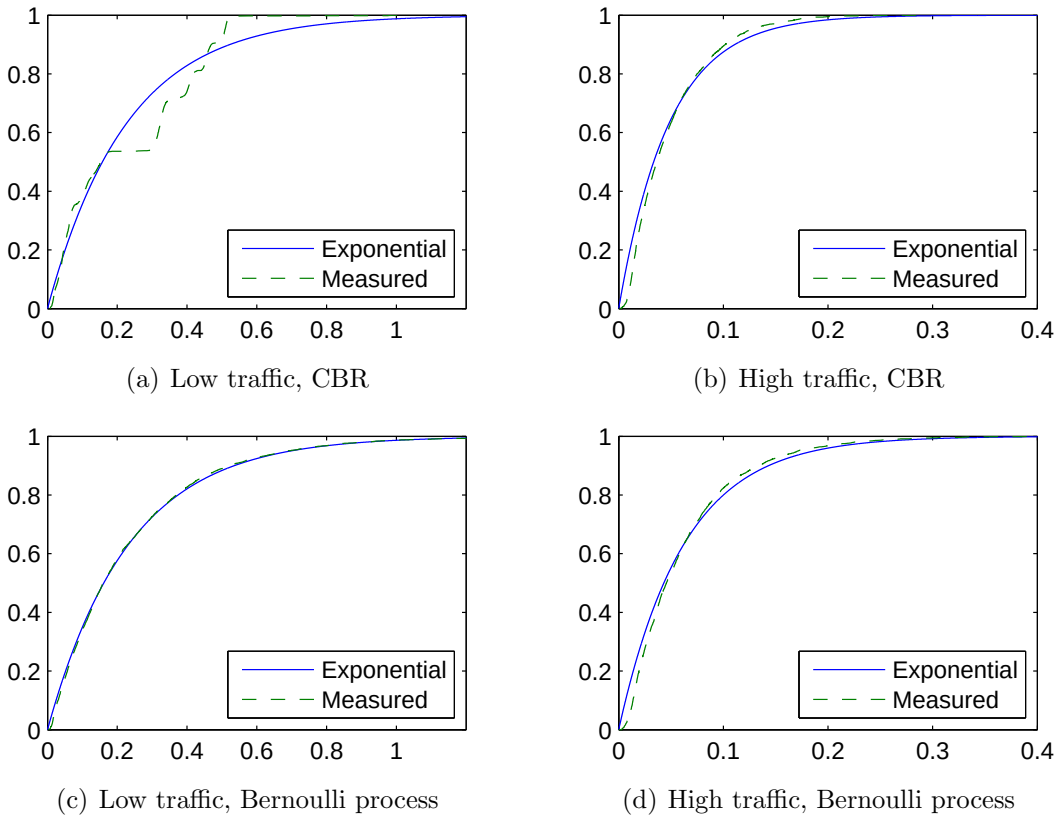
For event-based applications, nodes send data only if a certain physical event of interest occurs, e.g., the temperature exceeds a given threshold. We model the interarrival time of locally generated packets by the Geometric distribution. This is motivated by the following. In such applications, the generated data are often sporadic. Considering such physical events do not occur very frequently, the probability that the event occurs at any time is governed by a Poisson process, and the inter-arrival time is exponentially distributed. In a discrete time model with a small enough time unit T_{u}, the probability that more than one event occurs in a time unit is negligible. Thus, assuming only one event occurs in a time unit, a Bernoulli process is used to approximate the Poisson process in each time unit, according to the definition of the Bernoulli process [67, Ch. 6].

2.1.3.2 Locally Generated Traffic Pattern: Monitoring Applications

For *monitoring applications*, nodes repeatedly detect the physical environment using their sensors. Thus, the generated data is periodic. Accordingly, the locally generated traffic can be modeled using a constant bit rate (CBR) model, i.e., the inter-arrival time of locally generated traffic is a constant T_{e}.

2.1.3.3 Relay Traffic Pattern

For the relay traffic, we approximate the interarrival time distribution based on empirical measurements. Testbed experiments have been conducted to estimate the distribution of the inter-arrival time of packets in a 10-hop chain network for both types of applications, i.e., monitoring and event-based for low and high traffic rates.



(a) Low traffic, CBR

(b) High traffic, CBR

(c) Low traffic, Bernoulli process

(d) High traffic, Bernoulli process

Figure 2.1: The distribution of inter-arrival time for different types of traffic in a 10-hop chain network.

In each experiment, each node uses the TinyOS CSMA/CA MAC protocol and generates packets according to either a CBR model (monitoring), or a Bernoulli process in each time unit of 0.1 s (event-based). Each node transmits its generated packets and the received packets from its neighbors to the next node toward the end of the chain. The distribution of the inter-arrival time of the packets is recorded at the end of the chain. The following combinations of the locally generated traffic rate and pattern are examined:

(a) 0.4 pkt/s (low traffic) and CBR;

(b) 4 pkt/s (high traffic) and CBR;

(c) 0.4 pkt/s (low traffic) and Bernoulli process;

(d) 4 pkt/s (high traffic) and Bernoulli process.

The empirical *cdf* of the inter-arrival time is shown in Figure 2.1 along with an exponential distribution model for four cases[1]. The results reveal that except for the light periodic traffic case shown in Figure 2.1(a), exponential distribution closely models the inter-arrival rate. The light periodic traffic and other types of traffic, such as bursty traffic, can be captured by extending our queueing model to adopt a Markov Arrival Process (MAP) [69, Ch. 5], and are left as a future research topic. Accordingly, in our discrete-time model, we consider that the inter-arrival time for event-based applications follows a Geometric distribution, and define the traffic rate λ at some node to be the probability that a new locally generated packet or relay packet arrives during a time unit T_{u}.

2.2 Approaches

The analytical framework in this dissertation utilizes a bottom-up approach in two levels. First, at the node level, the single-hop delay distribution, the single-node energy consumption, and the single-node lifetime distribution are obtained. Then, the single-hop delay distribution and the single-node lifetime distribution are used to obtain the end-to-end delay distribution, the network lifetime distribution, and the event detection distribution at the network level. Finally, the distribution of end-to-end delay, network lifetime, and event detection delay are used as objective or constraint metrics in the probabilistic optimization framework.

[1]The exponential distribution shown in the figures are chosen such that their mean equals the measured inter-arrival times.

2.2.1 Node-Level Analysis

The framework at the node level is based on a Discrete-Time Markov Model. Each node is modeled according to a first-come-first-serve queuing model, which is characterized by its packet arrival process and service process. In the proposed model, time is divided into time units with duration T_{u}. The packet arrival process is characterized by a Bernoulli process in each time unit, as discussed in Section 2.1.3. Moreover, a Discrete Time Markov Process (DTMP) is used to model the service behavior of the protocol with time unit, T_{u}. Therefore, the service time is Phase-Type (PH) distributed [68]. Considering a single processor at each node and a queue capacity of M, the resulting model is a discrete time Geom/PH/1/M queueing model, and the system is essentially governed by a Quasi-Birth-Death (QBD) process [68].

A layered discrete-time recurrent Markov chain, $\{X_n\}$, is used to model the DTMP at each node, with states and transitions among states representing the node operations. The communication protocols of each node are represented by transitions among the states. This Markov chain $\{X_n\}$ is directly used to derive the energy consumption distribution for each node. Then, the single-hop delay distribution is obtained as the absorption time of $\{Y_n\}$, an absorbing variation of $\{X_n\}$. The detailed models for the derivation of node-level analysis will be presented in Chapters 3 and 5, respectively.

2.2.2 Network-Level Analysis

In the proposed analytical framework, when random node deployment is considered, the location of each individual node cannot be determined for analysis purpose. Thus, the interaction among nodes is intractable using traditional deterministic analysis approaches. Motivated by the fact that the individual node properties are insignificant

when network-level performance is concerned, the network is represented by a continuous fluid entity distributed in the entire network area. Accordingly, the complexity of the model can be greatly reduced. The entire network area is divided into very small area elements, and according to the density of the network and the size of the area element, each area element is treated as if it has a fraction (i.e., not necessarily an integer number) of nodes. Moreover, the single-hop delay distributions among these area elements are calculated, and are used to calculate end-to-end delay distributions; the single-node lifetime distributions for nodes in each of the area elements are used to calculate the network lifetime distribution.

Furthermore, when performance of services involving a group of packets are considered, the traffic streams are treated in a similar way. One example is the analysis of event detection delay, where multiple report packets must be received by the sink before the event is detected. In such scenarios, the delay of individual packets is only a part of the event detection delay. Thus, the traffic to the sink is not considered as individual packets, but continuous packet flows. The average fraction of packets transmitted during any given time period is obtained, and is used to calculate the event detection delay.

By utilizing this spatio-temporal fluid model, the spatial node distribution and temporal packet distribution are approximated by their respective average processes. As a result, the complexities of the problems in both spatial and temporal domains are reduced, and the problems become tractable. Note that essentially these approximations are to ignore the randomness in the concerned distributions. This seemingly contradicts our goal to analyze the QoS metrics for their probabilistic characteristics. However, these approximations are necessary to make the problems tractable. Moreover, we try to limit such approximations to the minimum. For example, in the probabilistic analysis of single packet delay and single-node lifetime, the temporal

fluid model is not used.

In Chapters 3, 4, and 5, the details of the spatio-temporal fluid model is provided. The accuracy of this model is validated by the testbed and simulation evaluations.

2.2.3 Multi-Initial-Point Global Optimization Technique

In the probabilistic QoS optimization problem, the QoS metrics can be the objective function or constraint functions. However, due to the generality of the network topology and communication protocols assumed for the analysis framework, without *a priori* knowledge of the topology and protocol, the probabilistic QoS metric functions cannot be considered convex, nor can they easily be converted to convex problems. Thus, solving the optimization problems is non-trivial. The following heuristic-based global optimization technique is developed to tackle this issue.

In our proposed solution to the problem, N_{search} local-optimum searches are conducted with random initial search points. In each of the multiple searches, the initial set of network parameters is determined by randomly choosing sets within the parameter space, until the set of parameters satisfies all constraints. Starting from this set, a derivative based local optimum search is conducted. Then, the global optimum is approximated as the best result in all the N_{search} optimum values found by each of the local searches. In the case when one or more of the local searches cannot converge due to non-convexity, the search procedures are terminated.

By utilizing this multiple local search technique, the problem is solved without prior knowledge about the topology and protocol. Moreover, the optimum found by this technique asymptotically is always the global optimum when N_{search} is large. By adjusting the value of N_{search}, a trade-off can be achieved between the accuracy of the result and search time.

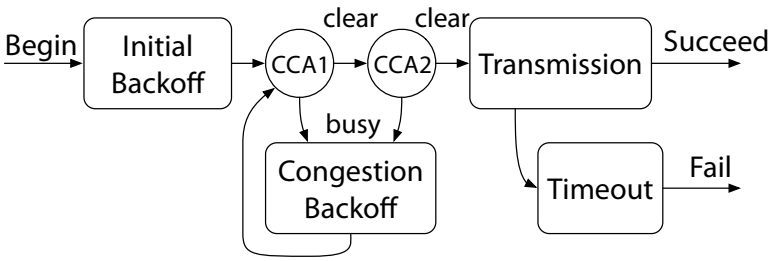


Figure 2.2: The transmission process for a packet with the TinyOS CSMA/CA MAC protocol.

This multi-initial-point global optimization technique is presented in Chapter 6.2.

2.3 Case Studies

Our proposed framework is designed to be generic and can be parameterized to analyze WSNs with practical MAC and routing protocols. To illustrate how the framework can be applied to the analysis for practical protocols, in this dissertation, two MAC protocols, i.e., the TinyOS CSMA/CA protocol (i.e., B-MAC [78] without Low Power Listening) and the Anycast protocol [49, 52, 58, 76, 95], are discussed as case studies in the analysis for each part in the analytical framework. Moreover, a basic example on a simple protocol is provided in Chapter 3 when the Discrete-Time Markov Model is first explained in detail. This example serves the purpose of explaining how the Markov model is constructed, and is not summarized here.

It should be noted that, in this dissertation, it is assumed that no in-network processing, such as data aggregations are employed. The probabilistic QoS analysis for WSNs with in-network processing could be a direction for future research. In the following, the two protocols are explained in detail.

2.3.1 TinyOS CSMA/CA MAC Protocol

The TinyOS default CSMA/CA protocol [91] is widely adopted by applications due to its simplicity and the popularity of TinyOS. The transmission process for a packet with this protocol is shown in Figure 2.2. When each node has a packet to send, a random *initial backoff* is conducted to arbitrate with other nodes. Then, similar to the IEEE 802.15.4 protocol [45], a two-slot Clear Channel Assessment (CCA) is performed, followed by the packet *transmission* if both CCAs detect the channel to be clear. If the channel is busy in either CCA, a random *congestion backoff* is conducted and the channel is sensed again. After the transmission is completed, the node waits for the acknowledgment from the receiver until ACK timeout.

If the acknowledgment is received, the packet is then transmitted successfully. Otherwise (ACK timeout), the transmission process is performed again beginning with the initial backoff. The process is repeated until either the transmission is successful, or a maximum number of transmission attempts, N_{tx}, is reached.

The TinyOS CSMA/CA protocol is a simple but representative CSMA/CA protocol. The protocol itself does not define any routing policy, and requires additional routing protocols. In this dissertation, for networks with the TinyOS CSMA/CA protocol, we focus on the routing protocols with static routing paths, or the steady-state period of dynamic routing policies, for example, the protocols that utilize the Geographic routing technique [106]. In such protocols, for a particular packet, a node forwards it to any of its neighbor nodes with a probability, which does not change rapidly over time.

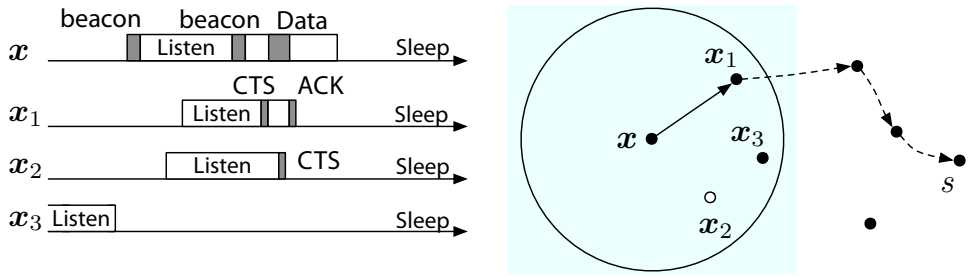


Figure 2.3: The transmission process and routing path for a packet with the anycast protocol.

2.3.2 Anycast Cross-Layer Protocol

To save communication energy, recent research has been focused on MAC protocols with duty cycle operations [11, 78]. In such protocols, nodes periodically enter active and sleeping states, and consume significantly less energy compared to nodes with MAC protocols that require the nodes to be always active. As a result of constantly entering a sleeping state, the communication delay is often increased. To counter this drawback, opportunistic routing techniques, particularly anycast protocols, are utilized along with a high node density to exploit node deployment redundancy [49, 52, 58, 76, 95]. The anycast technique is a cross-layer approach that exploits both temporal and spatial efficiency, with operations based on duty cycle sleeping and selective forwarding according to the location or the operation of neighbor nodes. With the anycast technique, if a node has packets to send, it first broadcasts a series of beacon messages. Then, one of the responding neighbors is chosen as the next-hop node according to predefined rules (e.g., the first node that responds, or the closest node to the destination). Finally, the sender forwards the data packet to the chosen neighbor.

While there is no single dominantly used anycast technique in WSNs, in this work, we investigate the following representative protocol (thereby referred to as the

"anycast protocol").

In the anycast protocol, sensor nodes report their readings to the sink, which is located at the center of the circular plane, through multi-hop routes in the network. The nodes (excluding the sink) turn off their radio periodically to save energy. When a node $\boldsymbol{x}$ has a packet to send, similar to the preamble packets in X-MAC [11], it starts to repeatedly transmit Request-to-Send (RTS) beacon packets based on a CSMA/CA manner, i.e., the channel is sensed before the beacon transmission. If the channel is busy, a random backoff is performed and the channel is sensed again. As shown in Figure 2.3, when any other node $\boldsymbol{x}'$ in the transmission range is awake and hears the packet, it checks for the following criteria: 1) node $\boldsymbol{x}'$ is closer to the sink than $\boldsymbol{x}$, and 2) the signal-to-noise ratio (SNR) of the received RTS packet, ψ, is greater than some predefined threshold ψ_{th}. If both criteria are met, node $\boldsymbol{x}'$ sends a Clear-to-Send (CTS) packet. Node $\boldsymbol{x}$ then chooses the first node that sends a CTS packet as the next-hop node and transmits the data packet to it. Successful data packet transmissions are acknowledged by the receiver, otherwise the sender retransmits the data packet until successful.

To reduce the waiting time for the packets spent in the queue and balance the energy consumption in the network, in the protocol each node responds to beacon packets only when it does not have packets to send. Considering the sink is awake all the time, if a node closer than a distance threshold r_{th} to the sink transmits beacons, it is assumed that no node except the sink will respond. Here r_{th} is chosen such that a high SNR is almost always guaranteed. Moreover, nodes go to sleep when they finish transmitting all packets in the queue. As a result, compared to non-transmitting nodes, the active period is shorter. In cases where transmission energy consumption is higher than listening, this helps balance energy consumption among nodes.

2.4 Testbed and Simulation Validations

The proposed analytical framework in this dissertation is validated extensively using both testbed experiments and simulations. Conducting testbed experiments for random deployment requires hundreds of realizations of the random topology before valid statistics can be gathered. Therefore, it is infeasible to validate our model *solely* using testbed experiments for random deployment. Simulations run much faster than testbed experiments, and can also run in multiple computers in parallel, thus reducing validation time in larger scale and longer duration. Hence, the testbed experiments are conducted for two purposes: to validate the proposed framework in a *realistic* setting, and to validate the accuracy of computer simulations.

2.4.1 Testbed Experiments

Our testbeds are located in the Cyber-Physical Networking Laboratory (CPN Lab) of UNL. Two different testbeds are used. The first one is a testbed in the ceiling of the CPN Lab, consisting of USB sockets connected to a central computer, as shown in Figure 2.4. Sensor nodes are connected to the USB sockets. The testbed supports various types of off-the-shelf sensor nodes, including TelosB motes and Mica series motes. The other testbed is a lightweight frame that is suspended in the CPN Lab, with strings and wooden sticks forming a grid, as shown in Figure 2.5. Sensor nodes can be hung below the frame using the grid as anchors. In both testbeds, Crossbow TelosB motes are used. They are placed in the testbed at specific locations according to the experiment requirements. The delay of communication and the energy consumption of nodes are measured using the techniques described below.

Figure 2.4: The testbed in the ceiling of the CPN Lab.

2.4.1.1 Measuring the Communication Delay

To measure the end-to-end communication delay from a source node to the destination node, programs running in the source node and destination node are modified. When the source node generates a packet, an electric pulse is simultaneously generated by the program in the source node, and is sent to the destination node through a pair of wires. The destination node starts a timer when it receives a pulse, and then waits for the corresponding packet. When the packet is received by the destination node, the duration after the reception of the pulse is recorded as the packet delay (the time spent on the signal transmission over the wires is ignored). This technique does not require destination nodes to know the clock information of the source nodes, and thus eliminates the need for synchronization among all the nodes.

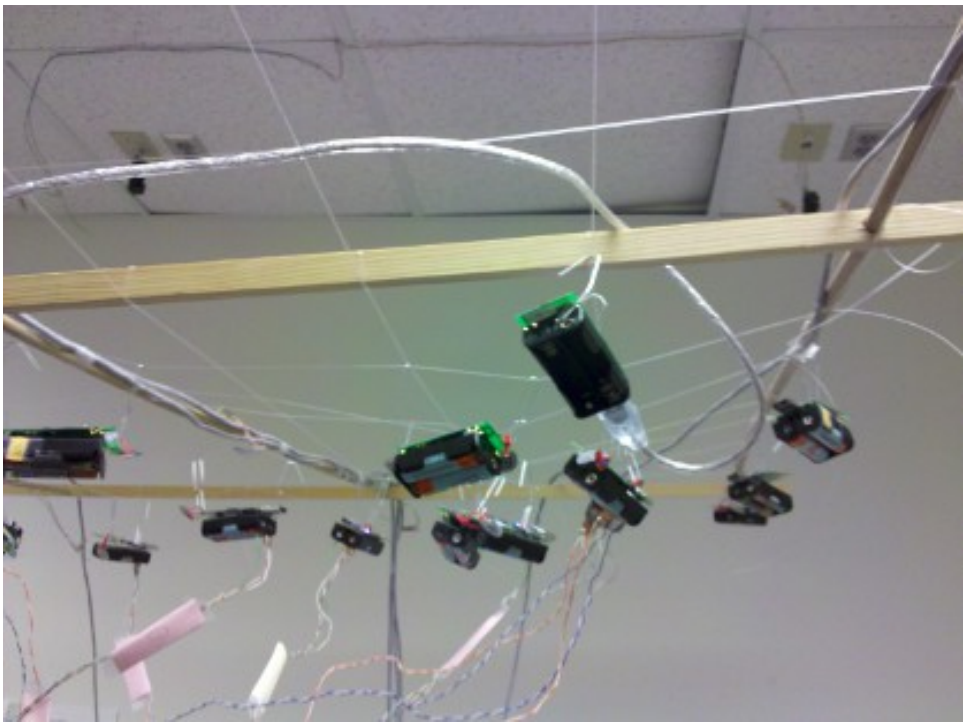


Figure 2.5: The suspended frame testbed in the CPN Lab.

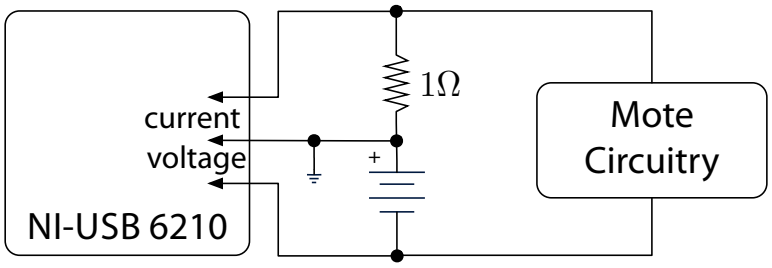


Figure 2.6: The technique used to measure the current drawn by each node. A 1 Ω resistor is placed in the circuit, and a NI-USB 6210 DAQ module is used to log the current and the voltage of the battery.

2.4.1.2 Measuring the Energy Consumption of Nodes

To measure the energy consumption of each node in a given period of time, the current drawn by each node is measured using NI-USB 6210 data acquisition (DAQ) modules [89]. In both testbeds, as shown in Figures 2.6 and 2.7 for each node, a 1Ω resistor is placed in the circuit loop, and the current drawn from the battery is obtained by

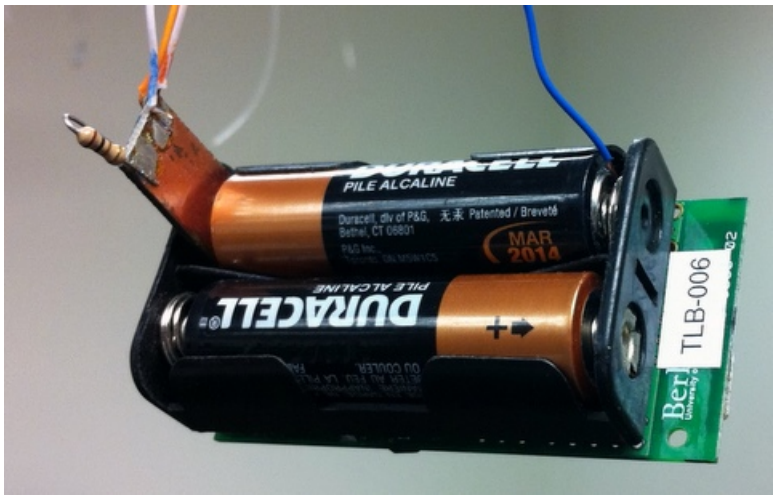


Figure 2.7: The energy monitoring circuitry. A small PCB board, on which a 1-Ohm resistor is soldered on, is used to insert the resistor into the note circuitry.

measuring the voltage drop over the resistor. The voltage drop is measured using DAQ modules at 10kHz, converted to the current, and logged for 24 hours. Since in practice, the energy charge and consumption is usually measured in terms of the product of the current and time duration, solely measuring the current is enough to estimate the energy consumption of nodes. However, the testbeds are also able to sample the battery voltage simultaneously, as shown in Figure 2.6. Logged readings of the current and the voltage for each node are sent to the central computer for data analysis.

2.4.2 Simulations

The computer simulations are performed using TOSSIM [56], a mote simulator based on TinyOS. Actual node programs can run in the TOSSIM simulation environment with little or no modifications. To obtain statistically valid results from simulations, a large number of simulation trials needs to be completed. This motivates us to run the simulations on FireFly [43], a supercomputer located at the Holland Computing

Center of University of Nebraska-Lincoln.

To speed up TOSSIM simulations and obtain the simulation results for lifetime-scale durations, several techniques are utilized. First, the different independent trials of simulations are conducted *in parallel* on different processing units of the supercomputer. Second, TOSSIM code is modified such that all *log and debug information is reduced*, except for the minimum necessary log of the energy consumption. This reduces the time spent on time-consuming I/O operations. Third, in some experiments that require very long simulation durations, the realistic channel model in TOSSIM is replaced by a *simplified channel model*. The detailed discussion on these techniques and the results are provided in Chapter 5.

In the following chapters, the probabilistic end-to-end delay, the probabilistic network lifetime, and the probabilistic event detection delay are analyzed in detail, followed by the probabilistic QoS optimization framework.

Chapter 3

End-to-End Delay Distribution

In this chapter, the end-to-end communication delay distribution is analyzed. The goal of this chapter is to provide a comprehensive analytical model for distribution of end-to-end delay in WSNs. Accordingly, a comprehensive and accurate cross-layer analysis framework is developed to characterize the end-to-end delay distribution in WSNs. The effects of heterogeneity in WSNs on latency is captured in terms of channel quality, transmit power, queue length, and communication protocols. The developed framework highlights the relationship between network parameters and the delay distribution in multi-hop WSNs.

In the following, the related work in this area is summarized in Section 3.1. In Section 3.2, the end-to-end delay distribution problem is formally defined, and an overview of the proposed Markovian model is provided. The detailed derivation of the single-hop delay distribution is described in Section 3.3, followed by the derivation of the end-to-end delay distribution in Section 3.4. Then, case studies for the CSMA/CA MAC protocol and the anycast protocol are provided in Section 3.5 and Section 3.6, respectively. Experimental results are provided in Section 3.7 to validate the developed model. Finally, the conclusions of this chapter is given in Section 3.8.

3.1 Related Work

The problem of probabilistic end-to-end delay analysis has attracted a large amount of research in recent years. The concept of Network Calculus [20] has been extended to support probabilistic delay *bounds* in [12, 32, 51, 81]. Network calculus and its probabilistic extensions are based on a min-plus algebra to provide traffic curves and service curves, which are deterministic (or statistical) bounds of traffic rate and service time, respectively. In these studies, the *worst case* performance bounds are analyzed. However, determining worst case bounds has limited applicability in WSNs for three reasons: First, because of the randomness in wireless communication and the low power nature of the communication links, worst case bounds do not exist in most practical scenarios. Second, the large variance in the end-to-end delay in WSNs results in loose bounds that cannot accurately characterize the delay distribution. Finally, most applications tolerate packet loss for a lower delay of higher priority packets since the efficiency of the system is improved. These motivate the need for *probabilistic delay analysis* rather than worst case bounds.

Moreover, work on real-time queueing theory [55, 101] combines real-time theory and queueing theory to provide stochastic models for unreliable networks. However, these models consider heavy traffic rate (usually saturation mode), which is not applicable for WSNs. The approach in this dissertation is similar to real-time queueing theory [55] in that a stochastic queuing model for the analysis is used. In contrast, the focus of this dissertation is not on the real-time scheduling problem, which has been discussed intensively in the literature [55, 57, 101]. Rather, it is aimed to provide an analytical tool to help develop communication solutions.

Recently, the delay distribution of MAC protocols has been analyzed in a large number of studies for wireless networks and WSNs, in particular. The access delay of

several MAC protocols has been investigated including IEEE 802.11b DCF protocol [8, 80, 87], IEEE 802.15.4 protocol [77, 79], and TDMA protocols [70]. However, in these studies, a broadcast network is considered, where each node can hear the transmission of each other. Moreover, in [8, 80, 87], saturated traffic is considered. Consequently, the multi-hop communication effects due to hidden node problems and the low traffic rate of WSNs cannot be captured.

The distribution of link layer retransmissions are modeled in [47]. While the distribution of the number of retransmissions is obtained, the transmission time is regarded as the same for each attempt. Hence, the resulting delay distribution model does not consider the uncertainty due to random backoffs of CSMA/CA protocols. In [98], the end-to-end delay distribution in a linear network is derived for homogeneous networks. However, this model assumes infinite queue length at each node, which may not be practical considering the resource constraints of sensor nodes. A probabilistic end-to-end delay and network lifetime analysis is given for WSNs performing data aggregation in [35], but with the assumption that packet transmission time is exponentially distributed. This assumption is inaccurate for most of the MAC protocols commonly in use. Finally, in [30, 36, 74], empirical measurements are used to provide probabilistic estimations for end-to-end delay. These solutions exploit on-the-fly measurements but do not provide analytical results. Before this dissertation, completely and accurately characterizing end-to-end delay in WSNs was still an open problem.

In the following, the probabilistic end-to-end delay analysis is provided by first defining the problems.

3.2 Problem Definition and System Model

As described in Chapter 2, in our analysis, two types of network deployments, random and deterministic deployments, are considered. In both cases, each node is indexed by its location $\boldsymbol{x}$. For a given network with a given MAC protocol and node parameters, we are interested in the following two problems:

1) What is the probability distribution function (*pdf*) of single-hop delay , $f_{\mathrm{sh}(\boldsymbol{x},\boldsymbol{y})}(t)$, between two nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ for a new arriving packet?

2) Given the single-hop delay distribution, what is the *pdf* of the end-to-end delay, $f_{\mathrm{e2e}(\boldsymbol{x},\boldsymbol{s})}(t)$, between a node $\boldsymbol{x}$ and a sink located at $\boldsymbol{s}$?

We consider a heterogeneous network for this analysis, where the heterogeneity is defined in terms of channel conditions, the packet error rate, PER, traffic rate, λ, queue length, M, maximum number of retransmission attempts, N_{tx}, and transmission power, P_{tx}, with appropriate subscripts indicating the different values for different nodes. In the following, we provide an overview of our solutions for the two problems above and the detailed descriptions are deferred to Sections 3.3-3.4.

3.2.1 Single-hop Delay Distribution

Each node is modeled according to a queuing model, which is characterized by its inter-arrival distribution and service process. More specifically, we model the traffic inter-arrival according to a Geometric distribution as explained in Chapter 2. Furthermore, a Discrete Time Markov Process (DTMP) is used to model the service behavior, as stated in Chapter 2. Therefore, the service time is Phase-Type (PH) distributed [68]. Considering a single processor at each node and a queue capacity of M, the resulting model is a discrete time Geom/PH/1/M queueing model.

The communication system at each node is modeled as a discrete-time recurrent

Markov chain, $\{X_n\}$. As shown in Figure 3.1(a), this DTMC has a layered structure. Each layer i contains the part of the chain where there are i packets in the queue. The communication behaviors of each node are represented by transitions among states in $\{X_n\}$. Then, a second DTMC, $\{Y_n\}$, which is the absorbing variant of $\{X_n\}$, is used to obtain the single-hop delay distribution. The detailed explanation of these DTMCs is provided in Section 3.3.

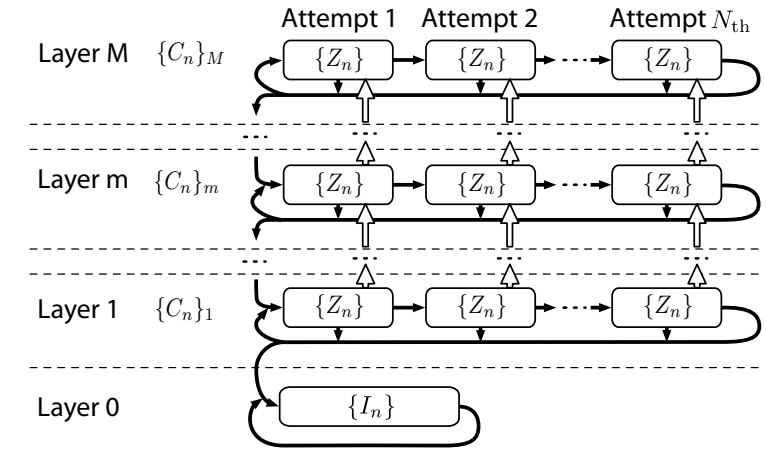
3.2.2 End-to-End Delay Distribution

With each hop modeled as a Geom/PH/1/M queue, the entire network is considered as a queueing network. Nodes are interrelated according to the traffic constraints. More specifically, the successfully transmitted traffic rate from one node should be equal to the sum of the incoming relay traffic rate at each of the next-hop neighbors of the node.

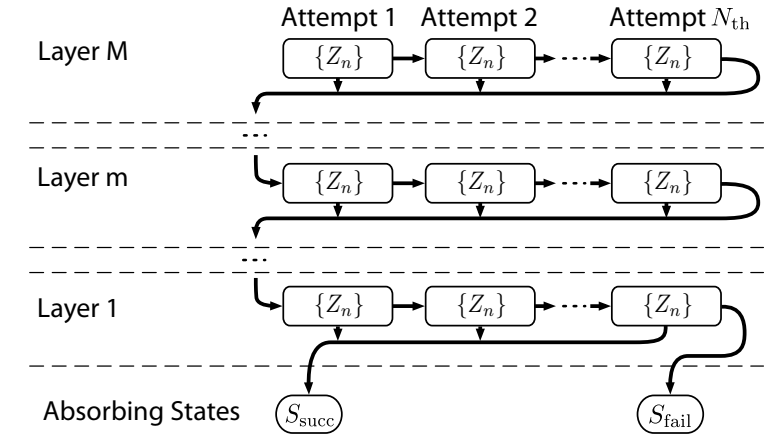
The topology of the queueing network depends on the routing protocol used. In this dissertation, we focus on the class of routing protocols with which each node maintains a probabilistic routing table for its neighbors, e.g., Geographic routing protocols [3]. Nodes relay their packets to each of their neighbors according to a probability in their routing tables. By first calculating the relay traffic and the single hop delay distribution for each pair of nodes, the end-to-end delay is obtained using an iterative procedure as will be explained in Section 3.4.

3.3 Single-hop Delay Distribution

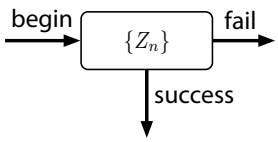
The communication system at each node is modeled by a DTMC $\{X_n\}$ and its absorbing variant $\{Y_n\}$. First, $\{X_n\}$ is constructed to capture the equilibrium behavior of the communication. Then, $\{Y_n\}$ is used to analyze the transient communication be-



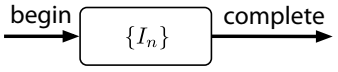
(a) $\{X_n\}$



(b) $\{Y_n\}$



(c) $\{Z_n\}$



(d) $\{I_n\}$

Figure 3.1: The structures of Markov chains (a) $\{X_n\}$ and (b) $\{Y_n\}$. Their building blocks are also shown: (c) $\{Z_n\}$ and (d) $\{I_n\}$

havior after a specific packet arrives. The single-hop delay of the packet transmission is then represented as the absorption time of $\{Y_n\}$. In the following, the construction of $\{X_n\}$ and $\{Y_n\}$ are described in detail, and the single-hop delay distribution is derived according to Theorem 1 at the end of this section.

3.3.1 Constructing Markov chain $\{X_n\}$

The DTMC, $\{X_n\}$, as shown in Figure 3.1(a), is composed of $M+1$ layers, where each layer m $(0 \leq m \leq M)$ represents the state where there are m packets in the queue and M is the queue capacity. These layers are of two different types, the *quiescent layer*, $\{I_n\}$, and the *communication layers*, $\{C_n\}_m$. The quiescent layer, $\{I_n\}$, $(m = 0)$ represents the *quiescent process*, during which the node does not have any packet to send, and waits for new packets. The communication layers, $\{C_n\}_m$ (m $>$ 0), represent the *communication process* in which packets are transmitted. One or more identical transmission attempts are conducted, until either the packet is successfully transmitted, or the maximum number of transmission attempts, N_{tx}, is exceeded. Accordingly, a layer m in $\{X_n\}$ is denoted by $\{C_n\}_m$, and is composed of N_{tx} blocks. The b-th block in layer m is denoted by $\{Z_n\}_{m,b}$[1]. As shown in Figure 3.1(c), each block models a single transmission attempt. The structure of $\{Z_n\}$ depends on the MAC protocol used. Packets are dropped if they arrive at a full queue or if all N_{tx} transmission attempts fail. Consequently, the v-th state in layer m and transmission attempt b is denoted by $S_{m,b,v}$.

The traffic arriving at each node contains locally generated traffic and relay traffic. While locally generated traffic can arrive at any time, the relay traffic can only arrive when the node is listening. Therefore, the total traffic rate depends on the state of the process. The locally generated traffic rate and the relay traffic rate for a node is denoted by λ_{lc} and λ_{re}, respectively. Therefore, in the states where the node is listening, the total traffic rate is $\lambda_{\text{lc}} + \lambda_{\text{re}}$, and it is λ_{lc} otherwise.

According to the MAC protocol employed, $\{I_n\}$ and $\{C_n\}$ are parameterized by the following notations:

[1]In the following, we drop the indices m and b, where appropriate, to simplify the notation

- $\boldsymbol{P}_I$ and $\boldsymbol{P}_C$: the transition probability matrices among the states in $\{I_n\}$ and $\{C_n\}$, respectively.
- $\boldsymbol{\alpha}_I$ and $\boldsymbol{\alpha}_C$: the initial probability vector for $\{I_n\}$ and $\{C_n\}$, respectively.
- $\boldsymbol{t}_I^{\mathrm{s}}$ and $\boldsymbol{t}_C^{\mathrm{s}}$: the probability vector from each state in $\{I_n\}$ and $\{C_n\}$ to complete the quiescent process and the communication process successfully, respectively.
- $\boldsymbol{t}_C^{\mathrm{f}}$: the probability vector from each state in $\{C_n\}$ to complete the communication process unsuccessfully.
- $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$: the packet arrival probability vector for each state in $\{I_n\}$ and $\{C_n\}$, respectively. Each element in the vectors equals to the probability of a new packet arrival in a time unit when the process is in the corresponding state.

The Markov chain block for each transmission attempt, $\{Z_n\}$, is characterized by the following:

- $\boldsymbol{P}_Z$, the transition probability matrix among the states in $\{Z_n\}$,
- $\boldsymbol{\alpha}_Z$, the initial probability vector for $\{Z_n\}$, and
- $\boldsymbol{t}_Z^{\mathrm{s}}$ and $\boldsymbol{t}_Z^{\mathrm{f}}$, the probability vector from each state in $\{Z_n\}$ to complete the transmission attempt successfully or unsuccessfully, respectively.

The states and the transitions related to $\{Z_n\}$ depend on the MAC protocol employed. For now, we assume that these matrices are known and the case studies to obtain them for two different protocols are provided in Section 3.5 and Section 3.6. Accordingly,

the transition probability matrix among the states in a single layer $\{C_n\}$ in $\{X_n\}$ is

$$\boldsymbol{P}_C = \begin{bmatrix} \boldsymbol{P}_Z & \boldsymbol{t}_Z^{\mathrm{f}}\boldsymbol{\alpha}_Z & & \boldsymbol{0} \\ & \ddots & \ddots & \\ & & \boldsymbol{P}_Z & \boldsymbol{t}_Z^{\mathrm{f}}\boldsymbol{\alpha}_Z \\ \boldsymbol{0} & & & \boldsymbol{P}_Z \end{bmatrix}, \tag{3.1}$$

where the number of $\boldsymbol{P}_Z$ blocks in $\boldsymbol{P}_C$ is equal to N_{tx}, i.e, the maximum number of attempts for each packet transmission. Similarly, the initial probability vector, $\boldsymbol{\alpha}_C$, and the probability vectors, $\boldsymbol{t}_C^{\mathrm{s}}$ and $\boldsymbol{t}_C^{\mathrm{f}}$ to complete a layer in success and failure are

$$\boldsymbol{\alpha}_C = \begin{bmatrix} \boldsymbol{\alpha}_Z & \boldsymbol{0} & \cdots & \boldsymbol{0} \end{bmatrix} \tag{3.2}$$

$$\boldsymbol{t}_C^{\mathrm{s}} = \begin{bmatrix} \boldsymbol{t}_Z^{\mathrm{s}} & \boldsymbol{t}_Z^{\mathrm{s}} & \cdots & \boldsymbol{t}_Z^{\mathrm{s}} \end{bmatrix}^{\mathrm{T}} \tag{3.3}$$

$$\boldsymbol{t}_C^{\mathrm{f}} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{t}_Z^{\mathrm{f}} \end{bmatrix}^{\mathrm{T}} \tag{3.4}$$

respectively.

The transition probability matrix, $\boldsymbol{Q}_X$, of the entire Markov chain $\{X_n\}$ can then be found according to transitions between different states at each layer as explained next.

For layer m, $1 \leq m \leq M-1$, the queue is not full. Whenever a packet arrives, the process transits to a higher layer since the queue length increases. The probabilities of such transitions are governed by the probability matrix

$$\boldsymbol{A}_{\mathrm{u}} = (\boldsymbol{1}\boldsymbol{\lambda}_C)^{\mathrm{T}} \otimes \boldsymbol{P}_C, \tag{3.5}$$

where $\boldsymbol{1}$ is a properly dimensioned matrix containing all 1's, and $\otimes$ is the entry-wise

product operator. $\boldsymbol{\lambda}_C$ and $\boldsymbol{P}_C$ are parameterized according to the MAC protocol. Note that element (v, v') in $\boldsymbol{A}_{\mathrm{u}}$ represents the transition probability from the v-th state in previous layer to the v'-th state in the upper layer, and other transition probability matrices in the following are defined the similar way. The transition probability matrix at the same level m, $1 \leq m \leq M-1$, is

$$\boldsymbol{A}_{\mathrm{s}} = (\mathbf{1}\boldsymbol{\lambda}_C)^{\mathrm{T}} \otimes (\boldsymbol{t}_C \boldsymbol{\alpha}_C) + (\mathbf{1} - \mathbf{1}\boldsymbol{\lambda}_C)^{\mathrm{T}} \otimes \boldsymbol{P}_C, \tag{3.6}$$

where $\boldsymbol{t}_C = \boldsymbol{t}_C^{\mathrm{s}} + \boldsymbol{t}_C^{\mathrm{f}}$ is the probability vector from each layer to complete the current communication process regardless of success or failure. The first term in (3.6) captures the case where a locally generated packet arrives at the same time unit in which a packet service is completed. The second term in (3.6) is for the case where neither service completion nor new packet arrival occurs during the time unit.

At layer $m = M$, the queue is full. Hence, new arriving packets are directly dropped. Therefore, the transition probability matrix in this layer is $\boldsymbol{A}_{\mathrm{u}} + \boldsymbol{A}_{\mathrm{s}}$.

When there is no packet arrival and the current packet service is completed, the Markov chain transits to one layer below. The transition probability matrix from level $m+1$ to level m, $1 \leq m \leq M-1$ is

$$\boldsymbol{A}_{\mathrm{d}} = (\mathbf{1} - \mathbf{1}\boldsymbol{\lambda}_C)^{\mathrm{T}} \otimes (\boldsymbol{t}_C \boldsymbol{\alpha}_C). \tag{3.7}$$

The transition probabilities are similar when the quiescent layer is involved as

shown below:

$$\boldsymbol{A}_{\mathrm{u}0} = \boldsymbol{\lambda}_I^{\mathrm{T}} \boldsymbol{\alpha}_C, \tag{3.8}$$

$$\boldsymbol{A}_{\mathrm{d}0} = (\mathbf{1} - \mathbf{1}\boldsymbol{\lambda}_C)^{\mathrm{T}} \otimes \boldsymbol{t}_C \boldsymbol{\alpha}_I, \tag{3.9}$$

$$\boldsymbol{A}_{\mathrm{s}0} = (\mathbf{1} - \mathbf{1}\boldsymbol{\lambda}_I)^{\mathrm{T}} \otimes (\boldsymbol{P}_I + \boldsymbol{t}_C^{\mathrm{s}} \boldsymbol{\alpha}_I). \tag{3.10}$$

When a new packet arrives while there is no packet in the system, the chain transits from the quiescent layer to layer 1 according to $\boldsymbol{A}_{\mathrm{u}0}$ in (3.8). When the service is completed for the only packet in the system and no new packet arrives, the chain transits from layer 1 to the quiescent layer according to $\boldsymbol{A}_{\mathrm{d}0}$ in (3.9). Finally, the transition probabilities with which the node stays in the quiescent layer are given in $\boldsymbol{A}_{\mathrm{s}0}$ in (3.10).

Using (3.5)-(3.10), the transition probability matrix $\boldsymbol{Q}_X$ for the entire recurrent Markov chain $\{X_n\}$, can be constructed as follows:

$$\boldsymbol{Q}_X = \begin{array}{c} \text{layer} \\ 0 \\ 1 \\ 2 \\ \cdots \\ M \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & \cdots & M \end{array} \\ \begin{pmatrix} \boldsymbol{A}_{\mathrm{s}0} & \boldsymbol{A}_{\mathrm{u}0} & & & \mathbf{0} \\ \boldsymbol{A}_{\mathrm{d}0} & \boldsymbol{A}_{\mathrm{s}} & \boldsymbol{A}_{\mathrm{u}} & & \\ & \boldsymbol{A}_{\mathrm{d}} & \ddots & \ddots & \\ & & \ddots & \boldsymbol{A}_{\mathrm{s}} & \boldsymbol{A}_{\mathrm{u}} \\ \mathbf{0} & & & \boldsymbol{A}_{\mathrm{d}} & \boldsymbol{A}_{\mathrm{s}} + \boldsymbol{A}_{\mathrm{u}} \end{pmatrix} \end{array}, \tag{3.11}$$

where each non-zero block corresponds to the transition probability among all layers. The duration of the time unit T_{u} is chosen to be small enough such that the probability of having two or more transitions in a single time unit is negligible. Therefore, it is only possible for $\{X_n\}$ to have intra-layer transitions and inter-layer transitions to

adjacent layers. Also note that the first row and column of blocks in $\boldsymbol{Q}_X$ corresponds to the transition probabilities from and to the quiescent layer, respectively. Then, the equilibrium state probability vector, $\boldsymbol{\pi}$, for $\{X_n\}$ is calculated by solving $\boldsymbol{\pi}\boldsymbol{Q}_X = \boldsymbol{\pi}$ and $\sum_i \boldsymbol{\pi}_i = 1$, as described in the following.

Denote $\boldsymbol{\pi}_m$ as the sub vector in $\boldsymbol{\pi}$ corresponding to the states in layer m. According to [67, Ch. 9]:

$$\boldsymbol{\pi}_m = \boldsymbol{\pi}_{m-1}\boldsymbol{R},\ 2 \leq m \leq M-1, \tag{3.12}$$

where $\boldsymbol{R}$ is a constant rate matrix. Since,

$$\boldsymbol{\pi}_{m-1}\boldsymbol{A}_{\mathrm{u}} + \boldsymbol{\pi}_m\boldsymbol{A}_{\mathrm{s}} + \boldsymbol{\pi}_{m+1}\boldsymbol{A}_{\mathrm{d}} = \boldsymbol{\pi}_m,\ 2 \leq m \leq M-1,$$

the following iterative computation is conducted to solve $\boldsymbol{R}$:

$$\boldsymbol{R}^{(n+1)} = -\boldsymbol{A}_{\mathrm{u}}(\boldsymbol{A}_{\mathrm{s}} - \boldsymbol{I})^{-1} - (\boldsymbol{R}^{(n)})^2\boldsymbol{A}_{\mathrm{d}}\boldsymbol{A}_{\mathrm{s}}^{-1}, \tag{3.13}$$

where $\boldsymbol{R}^{(0)} = \boldsymbol{0}$. The iteration continues until there is a negligible difference between $\boldsymbol{R}^{(n+1)}$ and $\boldsymbol{R}^{(n)}$. Consequently, considering $\boldsymbol{\pi}_m = \boldsymbol{\pi}_1\boldsymbol{R}^{m-1}, 2 \leq m \leq M-1$, $\boldsymbol{\pi}$ can be solved for $M \geq 2$, as follows:

$$\begin{cases} \boldsymbol{\pi}_0 + \boldsymbol{\pi}_1(\boldsymbol{I} - \boldsymbol{R}^{M-1})(\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{e} + \boldsymbol{\pi}_M\boldsymbol{e} = \boldsymbol{e}^{\mathrm{T}} \\ \boldsymbol{\pi}_0(\boldsymbol{A}_{\mathrm{s}0} - 1) + \boldsymbol{\pi}_1\boldsymbol{A}_{\mathrm{d}0} = \boldsymbol{0} \\ \boldsymbol{\pi}_0\boldsymbol{A}_{\mathrm{u}0} + \boldsymbol{\pi}_1(\boldsymbol{A}_{\mathrm{s}} - \boldsymbol{I} + \boldsymbol{R}\boldsymbol{A}_{\mathrm{d}}) = \boldsymbol{0} \\ \boldsymbol{\pi}_{M-1}\boldsymbol{A}_{\mathrm{u}} + \boldsymbol{\pi}_M(\boldsymbol{A}_{\mathrm{s}} + \boldsymbol{A}_{\mathrm{u}} - \boldsymbol{I}) = \boldsymbol{0} \end{cases} \tag{3.14}$$

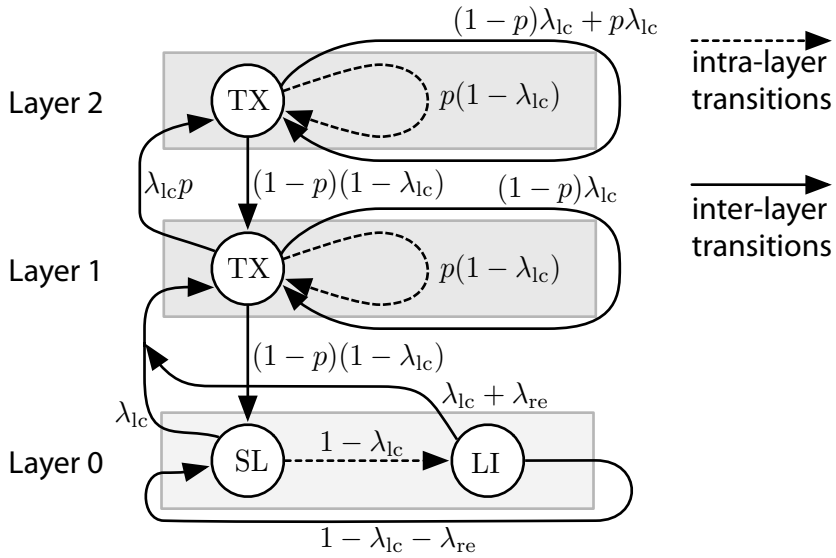


Figure 3.2: The structure of $\{X_n\}$ for the simple example.

and for $M = 1$, as follows:

$$\begin{cases} \boldsymbol{\pi}_0 + \boldsymbol{\pi}_1 \boldsymbol{e} = \boldsymbol{e}^{\rm T} \\ \boldsymbol{\pi}_0(\boldsymbol{A}_{\rm s0} - 1) + \boldsymbol{\pi}_1 \boldsymbol{A}_{\rm d0} = 0 \\ \boldsymbol{\pi}_0 \boldsymbol{A}_{\rm u0} + \boldsymbol{\pi}_1(\boldsymbol{A}_{\rm s} + \boldsymbol{A}_{\rm u} - \boldsymbol{I}) = \boldsymbol{0} \end{cases} \tag{3.15}$$

where $\boldsymbol{e}$ is a properly dimensioned column vector of all 1's.

3.3.2 A Basic Example

In the following, we show an example protocol to illustrate how the Markov chain $\{X_n\}$ is constructed. In the example protocol, a node conducts a duty cycle operation every 2 s. It first sleeps for 1 s and then listens on the channel for another 1 s. If a packet is received during the listening period with a probability $\lambda_{\rm re}$, or if a local packet is generated in any period with a probability of $\lambda_{\rm lc}$, the node attempts to transmit the packet. The transmission attempt takes 1 s with a failure rate p,

and the node persistently attempts to transmit the packet until successful. While transmitting, the node cannot receive any packets, but can still generate packets. The queue length is $M = 2$. For this protocol, a time unit of 1 s can be chosen since all time periods are 1 s. Then, the quiescent process can be modeled by two states, and the communication process can be modeled by one state, as shown in Figure 3.2. The quiescent process, $\{I_n\}$, contains a sleeping state (SL) and a listening state (LI), whereas the communication process, $\{C_n\}$, contains a single transmission state (TX). Accordingly, $\boldsymbol{P}_I, \boldsymbol{P}_C, \boldsymbol{\alpha}_I, \boldsymbol{\alpha}_C, \boldsymbol{t}_I^{\mathrm{s}}, \boldsymbol{t}_C^{\mathrm{s}}, \boldsymbol{t}_C^{\mathrm{f}}, \boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ are found as:

$$
\begin{aligned}
\boldsymbol{P}_I &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, & \boldsymbol{P}_C &= p, \\
\boldsymbol{\alpha}_I &= \begin{bmatrix} 1 & 0 \end{bmatrix}, & \boldsymbol{\alpha}_C &= 1, \\
\boldsymbol{t}_I^{\mathrm{s}} &= \begin{bmatrix} 0 & 1 \end{bmatrix}^{\mathrm{T}}, & \boldsymbol{t}_C^{\mathrm{s}} &= 1-p, & \boldsymbol{t}_C^{\mathrm{f}} &= 0, \\
\boldsymbol{\lambda}_I &= \begin{bmatrix} \lambda_{\mathrm{lc}} & \lambda_{\mathrm{lc}} + \lambda_{\mathrm{re}} \end{bmatrix}, & \boldsymbol{\lambda}_C &= \lambda_{\mathrm{lc}},
\end{aligned}
\tag{3.16}
$$

where $\boldsymbol{t}_C^{\mathrm{f}} = 0$ because the communication persistently attempts to transmit until successful, thus it can never fail. Therefore, the blocks in $\boldsymbol{Q}_X$ (see (3.11)) are expressed as

$$
\begin{aligned}
\boldsymbol{A}_{\mathrm{u}} &= \lambda_{\mathrm{lc}} p, & \boldsymbol{A}_{\mathrm{u0}} &= [\lambda_{\mathrm{lc}} \;\; \lambda_{\mathrm{lc}} + \lambda_{\mathrm{re}}]^{\mathrm{T}} \\
\boldsymbol{A}_{\mathrm{s}} &= \lambda_{\mathrm{lc}}(1-p) + (1-\lambda_{\mathrm{lc}})p, & \boldsymbol{A}_{\mathrm{s0}} &= \begin{bmatrix} 0 & 1-\lambda_{\mathrm{lc}} \\ 1-\lambda_{\mathrm{lc}}-\lambda_{\mathrm{re}} & 0 \end{bmatrix} \\
\boldsymbol{A}_{\mathrm{d}} &= (1-\lambda_{\mathrm{lc}})(1-p), & \boldsymbol{A}_{\mathrm{d0}} &= [(1-\lambda_{\mathrm{lc}})(1-p) \;\; 0]
\end{aligned}
\tag{3.17}
$$

Case studies for two practical protocols, the TinyOS CSMA/CA protocol and the anycast protocol, are provided in Section 3.5 and 3.6, respectively.

3.3.3 Absorbing time for $\{Y_n\}$

To obtain the distribution of single-hop delay for a packet, consider a particular packet that enters the system at time $t = t_0$. The single-hop delay of the packet is the time spent until it is transmitted or dropped. To derive the delay distribution, we use another DTMC, $\{Y_n\}$, as an absorbing variant of $\{X_n\}$. As shown in Figure 3.1(b), in $\{Y_n\}$, the quiescent layer of $\{X_n\}$ is replaced by two absorbing states S_{succ} and S_{fail}, corresponding to the two cases where the packet is successfully transmitted and dropped, respectively. In addition, all new packet arrivals are ignored since they do not interfere with the service time of the packet concerned. Thus, the state transitions occur only inside a layer or from layer $m+1$ to m. The steps to obtain $\{Y_n\}$ from $\{X_n\}$ is explained in the following.

Before the packet arrives, the system is in one of the states according to the equilibrium state probability vector, $\boldsymbol{\pi}$. After the new packet arrives, if the queue is full, the packet is immediately dropped. The probability of queue full is

$$p_{\text{qf}} = \boldsymbol{\pi}_M \boldsymbol{A}_{\text{u}} \mathbf{1}, \tag{3.18}$$

where $\boldsymbol{\pi}_M$ is the sub-vector in $\boldsymbol{\pi}$ corresponding to the M-th layer. Otherwise, the packet is inserted into the queue. The probability vector that the node is in a specific state after the new packet arrives is $\boldsymbol{\pi}' = \boldsymbol{\pi}\boldsymbol{Q}_Y^{\text{up}}$, where $\boldsymbol{Q}_Y^{\text{up}}$ is the transition probability matrix of $\{Y_n\}$ conditioned on the fact that the new packet arrives. $\boldsymbol{Q}_Y^{\text{up}}$ is derived from $\boldsymbol{Q}_X$ in (3.11) by replacing $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ with vectors of all 1's in (3.5)-(3.10) and replacing $\boldsymbol{A}_{\text{s}} + \boldsymbol{A}_{\text{u}}$ with $\boldsymbol{A}_{\text{s}}$. Note that $\boldsymbol{A}_{\text{u}}$ in the bottom-right block accounts for the

transition that will cause a packet to drop because of a full queue. Then, $\boldsymbol{\pi}'$ is the initial probability vector for $\{Y_n\}$.

Accordingly, the transition probability matrix for $\{Y_n\}$ is

$$\boldsymbol{Q}_Y = \begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \\ \boldsymbol{t}_Y^{\mathrm{s}} & \boldsymbol{t}_Y^{\mathrm{f}} & \boldsymbol{P}_Y \end{bmatrix}, \tag{3.19}$$

where the transition probabilities from and to the absorbing states S_{succ} and S_{fail} are listed in the first two rows and columns, respectively. The transition probability matrix among the transient states, i.e., all states except S_{succ} and S_{fail}, is given by

$$\boldsymbol{P}_Y = \begin{bmatrix} \boldsymbol{P}_C & & & \mathbf{0} \\ \boldsymbol{t}_C\boldsymbol{\alpha}_C & \boldsymbol{P}_C & & \\ & \ddots & \ddots & \\ \mathbf{0} & & \boldsymbol{t}_C\boldsymbol{\alpha}_C & \boldsymbol{P}_C \end{bmatrix}. \tag{3.20}$$

This is obtained from (3.11) by removing the first row and first column of blocks, and replacing $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ with vectors of all 0's in (3.5)-(3.10) for each remaining block. The transition probability vectors from each of the transient states to the absorbing states are

$$\boldsymbol{t}_Y^{\mathrm{s}} = \begin{bmatrix} \boldsymbol{t}_C^{\mathrm{s}} & \mathbf{0} & \mathbf{0} & \cdots \end{bmatrix}^{\mathrm{T}}, \tag{3.21}$$

$$\boldsymbol{t}_Y^{\mathrm{f}} = \begin{bmatrix} \boldsymbol{t}_C^{\mathrm{f}} & \mathbf{0} & \mathbf{0} & \cdots \end{bmatrix}^{\mathrm{T}}, \tag{3.22}$$

respectively, where $\boldsymbol{t}_C^{\mathrm{s}}$ and $\boldsymbol{t}_C^{\mathrm{f}}$ are given in (3.3) and (3.4), respectively. Finally, since a transition in $\{Y_n\}$ takes a time unit T_{u}, the following important results are directly

obtained:

Theorem 1. *The probability mass function (*pmf*) of the number of time units, K, a packet should wait before being transmitted and dropped are*

$$f_K^{\mathrm{s}}(k) = \boldsymbol{\alpha}_Y \boldsymbol{P}_Y^{k-1} \boldsymbol{t}_Y^{\mathrm{s}}, \tag{3.23}$$

$$f_K^{\mathrm{f}}(k) = \boldsymbol{\alpha}_Y \boldsymbol{P}_Y^{k-1} \boldsymbol{t}_Y^{\mathrm{f}}, \tag{3.24}$$

respectively, where $\boldsymbol{\alpha}_Y = (\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2, \cdots, \boldsymbol{\pi}'_M)$, i.e., $\boldsymbol{\pi}'$ without the elements corresponding to the quiescent layer, and $\boldsymbol{P}_Y^{k-1}$ represents the $(k-1)$-th power of $\boldsymbol{P}_Y$.

Proof. The theorem follows from [67, Ch. 9.5]. □

The *pmf* of the number of time units a packet should wait, regardless of being transmitted and dropped, is obtained by adding $f_K^{\mathrm{s}}(k)$ and $f_K^{\mathrm{f}}(k)$. Thus, the following corollary is directly obtained.

Corollary 1. *The* pmf *of single-hop delay, measured by the number of time units of T_{u}, is given by*

$$f_K(k) = \boldsymbol{\alpha}_Y \boldsymbol{P}_Y^{k-1} \boldsymbol{t}_Y. \tag{3.25}$$

Using this model, the probability that the packet is eventually delivered in success can also be found, and is given by the following corollary:

Corollary 2. *The delivery rate of a new arriving packet is*

$$p_{\mathrm{deli}} = \sum_{k=1}^{+\infty} f_K^{\mathrm{s}}(k) = \boldsymbol{\alpha}_Y (\boldsymbol{I} - \boldsymbol{P}_Y)^{-1} \boldsymbol{t}_Y^{\mathrm{s}} \ . \tag{3.26}$$

Of interest, the first two moments of the successful single-hop delay, which are widely used as the performance metrics in WSN applications, can also be derived.

Corollary 3. *The mean and variance of single-hop delay for a new arriving packet are*

$$\mu_K = \frac{\boldsymbol{\alpha}_Y(\boldsymbol{I}-\boldsymbol{P}_Y)^{-2}\boldsymbol{t}_Y^{\mathrm{s}}}{p_{\mathrm{deli}}}, \tag{3.27}$$

$$\sigma_K^2 = \frac{\boldsymbol{\alpha}_Y(2(\boldsymbol{I}-\boldsymbol{P}_Y)^{-3}-(\boldsymbol{I}-\boldsymbol{P}_Y)^{-2})\boldsymbol{t}_Y^{\mathrm{s}}}{p_{\mathrm{deli}}} - \mu_K^2, \tag{3.28}$$

respectively.

The derivations are straightforward since

$$\mu_K = \mathrm{E}[K] = \frac{\sum_{k=1}^{+\infty} k \cdot f_K^{\mathrm{s}}(k)}{\sum_{k=1}^{+\infty} f_K^{\mathrm{s}}(k)}, \tag{3.29}$$

$$\sigma_K^2 = \mathrm{E}[K^2] - (\mathrm{E}[K])^2 = \frac{\sum_{k=1}^{+\infty} k^2 \cdot f_K^{\mathrm{s}}(k)}{\sum_{k=1}^{+\infty} f_K^{\mathrm{s}}(k)} - \mu_K^2, \tag{3.30}$$

where E[] represents expectation.

Next, the end-to-end delay distribution based on the single-hop delay distribution analysis in this section is derived.

3.4 End-to-end Delay Distribution

The end-to-end delay distribution depends on the topology of the network and the routing protocol used. For both random and deterministic deployments, the steady state behavior of the routing protocol is considered. In the following, the end-to-end delay distribution for the deterministic deployment is provided first.

3.4.1 Deterministic Deployment

In a network with deterministic deployment, each node has a deterministic location, and the forwarding probabilities among nodes is determined with the knowledge of the locations. We consider a typical network setup for a common type of applications, where a single sink is used. In such a case, the network is viewed as a directed acyclic graph (DAG) [18]. Without loss of generality, this graph can be topologically sorted so that a node with a larger index never transmits a packet to a node with smaller index. In a network with N nodes, the sink node is denoted by index N.

Suppose in each time unit of T_{u}, each node i generates a local traffic of $\lambda_{\mathrm{lc},i}$ to the sink. Each packet is routed using a relay $k \in \mathbb{C}_i$ with probability $p^{\mathrm{fw}}_{i,k}$, where $\mathbb{C}_i$ is the set of potential relays from i to the sink. Thus, $\sum_{k\in\mathbb{C}_i} p^{\mathrm{fw}}_{i,k} = 1$, $\forall i$. First, the average relay traffic $\bar{\lambda}_{\mathrm{re},i}$ in each time unit from node i is calculated by solving the following equation system for every node:

$$\bar{\lambda}_{\mathrm{re},i} = \sum_{m=1}^{i-1} (\bar{\lambda}_{\mathrm{re},m} + \lambda_{\mathrm{lc},m}) p^{\mathrm{fw}}_{m,i}\, p_{\mathrm{deli},m,i},\ \forall i, \tag{3.31}$$

and $\bar{\lambda}_{\mathrm{re},1} = 0$, where $p_{\mathrm{deli},m,i}$ is the probability that a packet is successfully delivered from node m to i, as defined in (3.26). Then, since each node cannot receive packets in transmission and sleeping states, the relay traffic rate in the states, in which the node is capable to receive packets, is

$$\lambda_{\mathrm{re},i} = \bar{\lambda}_{\mathrm{re},i} / \pi^{\mathrm{listen}}_i, \tag{3.32}$$

where π^{listen}_i is the probability that i is in any state in which the node can receive packets, and is the sum of the probabilities corresponding to all such states in $\boldsymbol{\pi}_i$. Accordingly, the input traffic rate vectors $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ of a node i can be found according

to Section 3.3. Then, $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ are used in (3.5)-(3.10) to determine the single-hop delay distribution, $f_{\text{sh}(i,j)}(t)$, between a pair of nodes i and j as discussed in Section 3.3.

Finally, the end-to-end delay distribution is given as

$$f_{\text{e2e}(i)}(t) = \sum_{k=i+1}^{N-1} f_{\text{sh}(i,k)}(t) * f_{\text{e2e}(k)}(t) p_{i,k}^{\text{fw}} + f_{\text{sh}(i,N)}(t) p_{i,N}^{\text{fw}} \tag{3.33}$$

where $(*)$ is the convolution operator. Our testbed experiments show that it takes less than 2 minutes to obtain the end-to-end delay distribution between two nodes in a network consisting of 16 nodes with TinyOS CSMA/CA MAC protocol. This calculation time is affordable for protocol analysis.

For a more generic network, where there may be multiple sinks, the above procedure can be extended to derive the end-to-end delay distribution. Suppose in each time unit of T_{u}, each node i generates a local traffic of $\lambda_{\text{lc},i,j}$ to each destination j. Each packet from i is routed to the destination j using a relay $k \in \mathcal{N}_{i,j}$ with probability $p_{i,k,j}^{\text{fw}}$, where $\mathcal{N}_{i,j}$ is the set of potential relays from i to j. Thus, $\sum_{k \in \mathcal{N}_{i,j}} p_{i,k,j}^{\text{fw}} = 1$, $\forall i, j$. We first calculate the average relay traffic $\bar{\lambda}_{\text{re},i,j}$ in each time unit from node i to destination j by solving the following equation system for every pair of nodes:

$$\bar{\lambda}_{\text{re},i,j} = \sum_{m \in \mathcal{M}_{i,j}} (\bar{\lambda}_{m,j}^{\text{f}} + \lambda_{\text{lc},m,j}) p_{m,i,j}^{\text{fw}} \, p_{\text{deli},m,i}, \ \forall i, j \tag{3.34}$$

where $\mathcal{M}_{i,j}$ is the set of nodes that use i as the next hop to reach j, $\lambda_{\text{lc},m,j}$ is the locally generated traffic from m towards j, $p_{m,i,j}^{\text{fw}}$ is the probability that the routing policy chooses i as the next hop for a packet from m to j. Finally, $p_{\text{deli},m,i}$ is the probability that a packet is successfully delivered from node m to i, as defined in (3.26). Therefore, the overall average relay traffic rate of node i is found as $\bar{\lambda}_i^{\text{f}} = \sum_j \bar{\lambda}_{\text{re},i,j}$, and the relay

traffic rate in receiving-capable states is obtained by (3.32).

Then, the single-hop delay distribution, $f_{\mathrm{sh}(i,j)}(t)$, is obtained for each pair of nodes according to Section 3.3, and the end-to-end delay distribution, $f_{\mathrm{e2e}(i,j)}(t)$, between a node i and a sink j can be solved in an iterative way as follows:

$$\begin{aligned} f^{(0)}_{\mathrm{e2e}(i,j)}(t) =& f_{\mathrm{sh}(i,j)}(t), \\ f^{(n+1)}_{\mathrm{e2e}(i,j)}(t) =& \sum_{k \in \mathcal{N}_{i,j}} f_{\mathrm{sh}(i,k)}(t) * f^{(n)}_{\mathrm{e2e}(k,j)}(t) p^{\mathrm{fw}}_{i,k,j} + f_{\mathrm{sh}(i,j)}(t) p^{\mathrm{fw}}_{i,j,j}. \end{aligned} \tag{3.35}$$

The iteration process terminates when the difference between two consequent iterations is negligibly small.

3.4.2 Random Deployment

For the random deployment, the nodes are located in the network according to a Poisson point process with density ρ. The exact location for each node is stochastic because of the randomness. Therefore, geographic routing protocols [3] are often used due to their scalability and adaptability to the random geographic locations of the nodes. In such protocols, instead of the routing probability $p^{\mathrm{fw}}_{i,j}$ between any pair of nodes i and j, the routing probability between any pair of *locations* $\boldsymbol{x}$ and $\boldsymbol{y}$, $p^{\mathrm{fw}}_{\boldsymbol{x},\boldsymbol{y}}$ can be determined.

A common scenario is also considered for the random deployment, where the nodes in the network generate the same amount of local traffic to a sink. Moreover, each node $\boldsymbol{x}$ forwards packets to the neighboring nodes within its *feasible region* $\mathbb{F}_{\boldsymbol{x}}$, i.e., the region in which nodes are closer to the sink, but are still in the transmission range, as shown in Figure 3.3. Assume that the sink is located at the center of a circular plane with a radius R. In this scenario, the end-to-end delay analysis can take advantage of the symmetry of the topology as explained next.

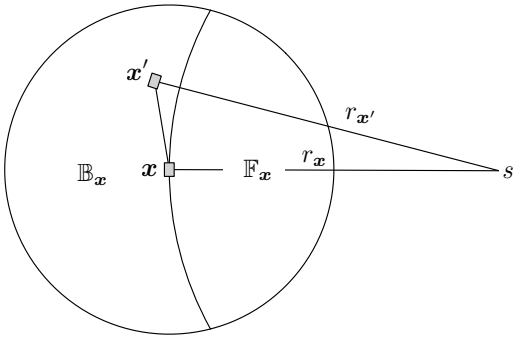


Figure 3.3: The feasible region, $\mathbb{F}_{\boldsymbol{x}}$, and the infeasible region, $\mathbb{B}_{\boldsymbol{x}}$, of node $\boldsymbol{x}$.

The entire circular plane is discretized into concentric rings indexed by their distance to the sink, r. Each node senses the physical events, and generates packets with traffic rate λ_{lc}. By symmetry, the relay traffic $\lambda_{\text{re},r}$ is the same for all nodes in the same ring r. In the following analysis, we assume a polar coordinates system with the sink located at the origin.

As shown in Figure 3.3, for a node $\boldsymbol{x}$ located at $\boldsymbol{x} = (r_{\boldsymbol{x}}, \theta_{\boldsymbol{x}})$, the relay traffic arrives from any node $\boldsymbol{y}$ in the *infeasible region* $\mathbb{B}_{\boldsymbol{x}} = \mathbb{C}_{\boldsymbol{x}} \setminus \mathbb{F}_{\boldsymbol{x}}$, i.e., the region in which nodes are farther to the sink but are still in the transmission range. To derive the relay traffic rate for $\boldsymbol{x}$ and other nodes in ring $r_{\boldsymbol{x}}$, consider the small area $(r_{\boldsymbol{x}} : r_{\boldsymbol{x}} + \Delta r, \theta : \theta + \Delta\theta)$ around node $\boldsymbol{x}$ located at $(r_{\boldsymbol{x}}, \theta)$. Similar to the deterministic deployment, the relay traffic rate $\lambda_{\text{re},r_{\boldsymbol{x}}}$ is given by

$$\begin{aligned}
\lambda_{\text{re},r_{\boldsymbol{x}}} &= \bar{\lambda}_{\text{re},r_{\boldsymbol{x}}} / \pi^{\text{listen}}_{r_{\boldsymbol{x}}}, \\
\bar{\lambda}_{\text{re},r_{\boldsymbol{x}}} &= \frac{\int_{\mathbb{B}_{\boldsymbol{x}}} \rho(\bar{\lambda}^{\text{f}}_{\boldsymbol{y}} + \lambda_{\text{lc}}) p^{\text{fw}}_{\boldsymbol{y},\boldsymbol{x}}\, p_{\text{deli},\boldsymbol{y},\boldsymbol{x}} \mathrm{d}\boldsymbol{y}}{\rho \Delta r \Delta\theta r_{\boldsymbol{x}}},
\end{aligned} \tag{3.36}$$

where ρ is the network density of the Poisson node distribution, $p^{\text{fw}}_{\boldsymbol{y},\boldsymbol{x}}$ and $p_{\text{deli},\boldsymbol{y},\boldsymbol{x}}$ are similarly defined as $p^{\text{fw}}_{m,i}$ and $p_{\text{deli},m,i}$ in (3.31), except that the nodes are indexed by

their locations.

Finally, $p^{\text{fw}}_{\boldsymbol{y},\boldsymbol{x}}$ in (3.36) is the routing protocol-specific probability that the node at $\boldsymbol{y}$ transmit packets to a node at $\boldsymbol{x}$. A case study for the anycast protocol will be provided in Section 3.6 to show how this probability is obtained.

Thus, according to (3.36), the traffic rate of node $\boldsymbol{x}$ at each state is determined. Accordingly, the input traffic rate vectors $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$ of node $\boldsymbol{x}$ can be found according to Section 3.3. Then, the equilibrium state probability for the DTMC $\{X_n\}$, $\boldsymbol{\pi}_{r_{\boldsymbol{x}}}$ is obtained. Note that in (3.36), the traffic rate for nodes in ring $r_{\boldsymbol{x}}$ depends on the traffic rate and delivery rate for nodes in their infeasible region. Therefore, the single-hop delay distribution is obtained first for nodes in the outmost ring, and then the inner rings in the decreasing order of the ring radius.

By symmetry, the end-to-end delay distribution to the sink is the same for all nodes with a same distance $r_{\boldsymbol{x}}$ to the sink, and is obtained by

$$f_{\text{e2e}(r_{\boldsymbol{x}})}(t) = \int_{\mathbb{F}_{\boldsymbol{x}}} p^{\text{fw}}_{\boldsymbol{x},\boldsymbol{y}} f_{\text{sh}(r_{\boldsymbol{x}})} * f_{\text{e2e}(r_{\boldsymbol{y}})}(t) \mathrm{d}\boldsymbol{y}. \tag{3.37}$$

The end-to-end delay distribution is found in the ascending order of the distance to the sink.

Next, in Section 3.5, the TinyOS CSMA protocol is used as a case study to show how the DTMCs, specifically, the single transmission attempt block $\{Z_n\}$, are constructed, and how the end-to-end delay distribution is obtained, in a deterministic deployed network. Likewise, another case study of the anycast protocol is provided to illustrate the end-to-end delay analysis in a randomly deployed network in Section 3.6.

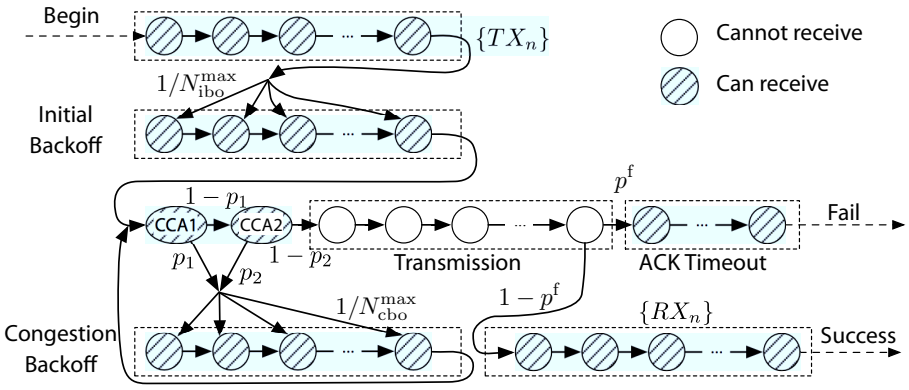


Figure 3.4: Markov chain structure for each attempt for TinyOS CSMA protocol. $N_{\text{ibo}}^{\max}$ and $N_{\text{cbo}}^{\max}$ are the number of states representing the initial backoff and congestion backoff, respectively.

3.5 Case Study: TinyOS CSMA/CA protocol

In this section, we illustrate how the single-hop delay distribution can be obtained for a particular MAC protocol in a deterministically deployed network. We use the TinyOS default CSMA/CA protocol [91], as described in Section 2.3.1. Several existing studies characterizing the CSMA/CA protocol in a broadcast network are discussed in Section 3.1. In this section, we refer to the framework in [79] for our analysis. Since multi-hop traffic and the hidden node problem are not considered in [79], we extend this analysis to the multi-hop case. Note that our aim in this section is not to propose yet another analysis of the CSMA/CA protocol. Instead, we illustrate how the existing models of MAC protocols can be extended through our framework to model the *end-to-end* delay distribution.

3.5.1 Markov Process Overview

With the TinyOS CSMA/CA protocol, nodes can start transmission at any time when a packet arrives. Therefore, the quiescent layer $\{I_n\}$ contains only one state, denoted here as S_{idle}. Moreover, the Markov chain, $\{Z_n\}$, that models each transmis-

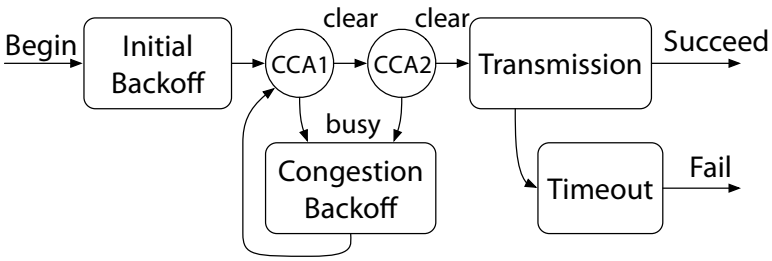


Figure 3.5: The transmission process for a packet with the TinyOS CSMA/CA MAC protocol. Figure 2.2 is redrawn here fore convenience.

sion attempt is depicted in Figure 3.4. Before each transmission, the packet in the queue is transferred from the microcontroller to the transceiver. The time needed for such transfer differs for various transceivers but is not negligible. Our experiments with TelosB nodes suggest that the durations of loading time before and after radio transmission are constant and are approximately 1.7 ms and 2.0 ms, respectively. Therefore, the data transfer delay is modeled by two additional state chains with a length corresponding to the transfer duration. These chains are the first and the last part of $\{Z_n\}$, denoted by $\{TX_n\}$ and $\{RX_n\}$ in Figure 3.4, respectively.

Other parts of $\{Z_n\}$, including the initial backoff, the congestion backoff, the CCAs, the packet transmission, and the ACK timeout are constructed according to Figure 3.5 (Figure 2.2 is redrawn here fore convenience).

3.5.2 Constructing the DTMC $\{X_n\}$

For each transmission attempt, the corresponding block of the Markov chain is depicted in Figure 3.4, which is characterized by three variables in the chain: p_1 and p_2 are the probabilities that the node senses the channel busy in the first and second CCA, respectively and $p_{\boldsymbol{x}}^{\mathrm{f}}$ is the probability that a transmission attempt fails due to either channel noise or collisions. For the derivations of their values, we first define the *collision area*, $\mathbb{C}_{\boldsymbol{x}}$, of a node $\boldsymbol{x}$ as the area in which all the neighbors interfere with

node $\boldsymbol{x}$. For two communicating nodes $\boldsymbol{x}$ and $\boldsymbol{y}$, both nodes reside in the intersection of the collision areas of these nodes, i.e., $\{\boldsymbol{x}, \boldsymbol{y}\} \in \mathbb{C}_{\boldsymbol{x},\boldsymbol{y}}$, where $\mathbb{C}_{\boldsymbol{x},\boldsymbol{y}} = \mathbb{C}_{\boldsymbol{x}} \cap \mathbb{C}_{\boldsymbol{y}}$. Moreover, the collision area of $\boldsymbol{x}$ that is not in $\mathbb{C}_{\boldsymbol{x},\boldsymbol{y}}$ is defined as $\mathbb{H}_{\boldsymbol{x},\boldsymbol{y}} = \mathbb{C}_{\boldsymbol{y}} \setminus \mathbb{C}_{\boldsymbol{x},\boldsymbol{y}}$, which is the *hidden node area* of $\boldsymbol{x}$ with respect to $\boldsymbol{y}$. Essentially, nodes that reside in $\mathbb{H}_{\boldsymbol{x},\boldsymbol{y}}$ cannot be heard by $\boldsymbol{y}$. Similarly, the hidden node area of $\boldsymbol{y}$ w.r.t. $\boldsymbol{x}$ is denoted by $\mathbb{H}_{\boldsymbol{y},\boldsymbol{x}}$.[2] The size of these areas $|\mathbb{C}_{\boldsymbol{x},\boldsymbol{y}}|$, $|\mathbb{H}_{\boldsymbol{x},\boldsymbol{y}}|$, and $|\mathbb{H}_{\boldsymbol{y},\boldsymbol{x}}|$ can easily be obtained according to the distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ and their respective interference ranges. Accordingly, the number of nodes in these areas are the product of their respective sizes and the network density ρ.

Denote $\phi_{\boldsymbol{x}}$ as the probability that node $\boldsymbol{x}$ is in the first CCA state. It is given by the sum of all probability elements corresponding to the first CCA states in $\boldsymbol{\pi}$. Note that since heterogeneous network traffic is considered, $\phi_{\boldsymbol{x}}$ may be different for different nodes. Also denote p_1 and p_2 as the probabilities that the node senses the first and the second CCA busy, respectively. Finally, denote $PER_{\boldsymbol{x},\boldsymbol{y}}$ as the packet error rate dependent on channel noise, which depends on the transmission distance, transmission power, random multi-path and shadowing effects. In our model, we define the expected packet reception rate for a pair of nodes according to the log-normal fading model in [107].

Then, the values of p_1, p_2 and p^{f} for each node are found by solving the following

[2]With a slight abuse of notation, in the following, we exclude the nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ from the nodes in these areas.

set of equations.

$$p_1 = p_{\text{send},\mathbb{C}_{\boldsymbol{x}}} L_{\text{TX}} + p_{\text{ack}} L_{\text{ACK}}, \tag{3.38}$$

$$p_2 = \left[1 - \frac{2 - p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}}}{2 - p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}} + \frac{1}{1-\prod_{k\in\mathbb{C}_{\boldsymbol{x}}}(1-\phi_k)}}\right](1 - \prod_{\boldsymbol{z}\in\mathbb{C}_{\boldsymbol{x}}}(1-\phi_{\boldsymbol{z}})) + \frac{1 - p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}}}{2 - p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}} + \frac{1}{\prod_{\boldsymbol{z}\in\mathbb{C}_{\boldsymbol{x}}}(1-\phi_{\boldsymbol{z}})}}, \tag{3.39}$$

$$p^{\text{f}}_{\boldsymbol{x},\boldsymbol{y}} = 1 - \frac{p^{\text{w}}_{\boldsymbol{x},\boldsymbol{y}}(1 - p^{\text{coll}}_{\boldsymbol{x},\boldsymbol{y}})(1 - PER_{\boldsymbol{x},\boldsymbol{y}})}{\phi_{\boldsymbol{x}}(1-p_1)(1-p_2)}, \tag{3.40}$$

where $p_{\text{send},\mathbb{C}_{\boldsymbol{x}}}$ is the probability with which *at least one* node $\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{x}}$ begins a transmission, p_{ack} is the probability that an ACK packet is transmitted by at least one node in $\mathbb{C}_{\boldsymbol{x}}$ during a time unit, $p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}}$ is the probability that a collision is observed on the channel on the condition that a transmission was going on, $p^{\text{w}}_{\boldsymbol{x},\boldsymbol{y}}$ is the probability that only node $\boldsymbol{x}$ starts to transmit a packet in the communication range of $\boldsymbol{y}$, and $p^{\text{coll}}_{\boldsymbol{x},\boldsymbol{y}}$ is the probability of collision due to hidden terminal transmissions. They are obtained as follows.

In (3.38), $p_{\text{send},\mathbb{C}_{\boldsymbol{x}}}$, the probability that at least one node $\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{x}}$ begins a transmission, is given by

$$p_{\text{send},\mathbb{C}_{\boldsymbol{x}}} = (1-p_1)(1-p_2)\left(1 - \prod_{\boldsymbol{z}\in\mathbb{C}_{\boldsymbol{x}}}(1-\phi_{\boldsymbol{z}})\right). \tag{3.41}$$

p_{ack}, the probability that an ACK packet is transmitted by at least one node in $\mathbb{C}_{\boldsymbol{x}}$ during a time unit, depends on the number of successful transmissions targeted into $\mathbb{C}_{\boldsymbol{x}}$ and is not trivial to determine. Motivated by the fact that the traffic rate and channel conditions do not change dramatically within a small area in most WSN applications, p_{ack} is approximated by the average probability of successful transmissions

from inside $\mathbb{C}_{\boldsymbol{x}}$. Thus,

$$p_{\text{ack}} = \sum_{\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{x}}} p^{\text{w}}_{\boldsymbol{z},\boldsymbol{x}} (1 - p^{\text{coll}}_{\boldsymbol{z},\boldsymbol{x}})(1 - PER_{\boldsymbol{z},\boldsymbol{x}}). \tag{3.42}$$

Then, in (3.39), $p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}}$, the probability that a collision is observed on the channel on the condition that a transmission was going on, is given by

$$p^{\text{nc}}_{\mathbb{C}_{\boldsymbol{x}}} = 1 - \frac{\sum_{\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{x}}} p^{\text{w}}_{\boldsymbol{z},\boldsymbol{x}}}{p_{\text{send},\mathbb{C}_{\boldsymbol{x}}}}. \tag{3.43}$$

Finally, in (3.40), (3.42), and (3.43), $p^{\text{w}}_{\boldsymbol{z},\boldsymbol{x}}$ is found by considering that no node $\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{y}}$ other than node $\boldsymbol{x}$ starts to transmit as follows:

$$p^{\text{w}}_{\boldsymbol{x},\boldsymbol{y}} = \phi_{\boldsymbol{x}}(1 - p_1)(1 - p_2) \prod_{\boldsymbol{z} \in \mathbb{C}_{\boldsymbol{y}}} (1 - \phi_{\boldsymbol{z}}). \tag{3.44}$$

Note that $p^{\text{f}}_{\boldsymbol{x},\boldsymbol{y}}$ is averaged among all destinations, $\boldsymbol{y}$, as the approximation of $p^{\text{f}}_{\boldsymbol{x}}$ for each node $\boldsymbol{x}$. As suggested in (3.40), the value of $p^{\text{f}}_{\boldsymbol{x},\boldsymbol{y}}$ depends on the channel conditions and the collision probability. Considering a channel-aware routing protocol is employed, $p^{\text{f}}_{\boldsymbol{x},\boldsymbol{y}}$ does not vary significantly for different node pairs and such approximation is acceptable. Accordingly, for a given node $\boldsymbol{x}$, the failure probability for each transmission attempt, $p^{\text{f}}_{\boldsymbol{x}}$, is the same for all packets in the queue.

The three probability values, p_1, p_2, and $p^{\text{f}}_{\boldsymbol{x}}$ are then used to construct the Markov chain, $\{Z_n\}$. Each of these values depends on each other as well as $\phi_{\boldsymbol{x}}$, which is the probability that the node $\boldsymbol{x}$ is in the first CCA state. Note that $\phi_{\boldsymbol{x}}$, p_1 and p_2 cannot be determined without the knowledge of $\boldsymbol{\pi}$, which can only be obtained after constructing the Markov chain as explained in Section 3.3. Consequently, an iterative procedure is used to find these parameters. First, an initial guess of $\phi_{\boldsymbol{x}}$, p_1 and p_2 is

used to construct the Markov chains for each node, based on which $\boldsymbol{\pi}$ is calculated. Then, values for $\phi_{\boldsymbol{x}}$, p_1 and p_2 are updated accordingly to the knowledge of $\boldsymbol{\pi}$. The calculation of $\phi_{\boldsymbol{x}}$, p_1, p_2, and $\boldsymbol{\pi}$ is conducted iteratively, until the difference of the value for any variable between two iterations is negligible.

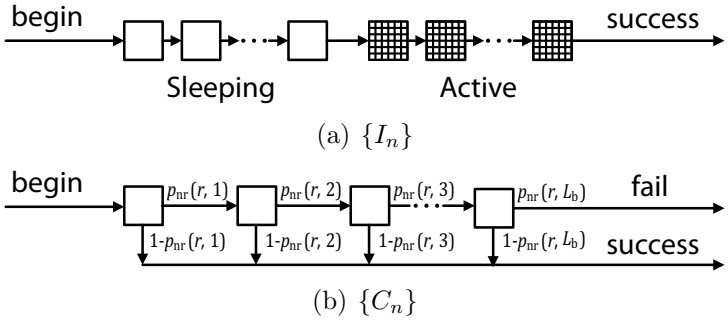
Accordingly, $\{Z_n\}$ is characterized by:

- The (v, v')-th element in $\boldsymbol{P}_Z$ is the transition probability from state v to v' shown in Figure 3.4. The transition probabilities p_1, p_2, and p^{f} are given by (3.38), (3.39) and (3.40), respectively. Other unnoted transition probabilities are 1.
- The element in $\boldsymbol{\alpha}_Z$ is 1 for states pointed by a "begin" arrow. Other elements are 0's.
- The element in $\boldsymbol{t}_C^{\mathrm{s}}$, and $\boldsymbol{t}_C^{\mathrm{f}}$ is set according to the probability attached to the arrows pointing to "success" and "fail", respectively.
- The elements in $\boldsymbol{\lambda}_Z$ corresponding to the states, which are denoted by "Can receive" in Figure 3.4, are set to $\lambda_{\mathrm{lc}} + \lambda_{\mathrm{re}}$. Other elements in $\boldsymbol{\lambda}_Z$ are set to λ_{lc}.

Moreover, $\{I_n\}$ has a single state S_{idle}, and is characterized by:

- $\boldsymbol{P}_I$ is a 1×1 matrix with the single element being 0.
- The single element in $\boldsymbol{\alpha}_I$ is 1.
- The single element in $\boldsymbol{t}_I^{\mathrm{s}}$ is 1.
- The single element in $\boldsymbol{\lambda}_I$ is set to $\lambda_{\mathrm{lc}} + \lambda_{\mathrm{re}}$.

After $\{Z_n\}$ and $\{I_n\}$ are constructed, the entire DTMC $\{X_n\}$ is obtained according to Section 3.3. The single-hop delay distribution is then derived by Theorem 1. Finally, the end-to-end delay distribution is found according to (3.33).



(a) $\{I_n\}$

(b) $\{C_n\}$

Figure 3.6: The Markov chain structure of (a) the communication process, $\{C_n\}$, and (b) the quiescent process, $\{I_n\}$, for the anycast protocol.

3.6 Case Study: Anycast protocol

In this section, the approach for computing single-hop and end-to-end delay distributions is illustrated for an anycast protocol, which is described in Chapter 2.3.2. This case study is used to show how the single-hop and the end-to-end delay analysis in Section 3.3 and Section 3.4 can be applied to protocols with *duty cycle* operations for a randomly deployed network. Other anycast protocols, and more generally, other duty cycle-based protocols, can be modeled using similar approaches.

For the random deployment of nodes, the topology model in Section 3.4.2 is considered, and node-specific variables are indexed by the ring radius r. In the following analysis, when there is no ambiguity, the subscript r in ring-specific variables is omitted.

We first show the DTMC $\{X_n\}$ for the protocol. Then, the protocol-specific parameters for the generic analysis in Section 3.3, including the relay traffic rate at each state, and the transition probabilities for $\{X_n\}$ are derived. The single-hop delay distribution for each pair of nodes is obtained after these parameters are known. Finally, the end-to-end delay distribution from each node to the sink is provided.

3.6.1 Markov Process Overview

The structures of $\{I_n\}$ and $\{C_n\}$ in DTMC $\{X_n\}$ for this protocol are shown in Figure 3.6. The quiescent layer $\{I_n\}$ consists of a chain of sleeping states and a chain of listening states of duration T_{u}. One may think that a single sleeping state and a single listening state are enough to model the duty cycle operation, similar to the basic protocol in Section 3.3.2. However, because of the memoryless nature of Markov process, arbitrary values of duty cycle must be captured with a specific number of states representing the active period and sleeping period.

When there is no communication, the Markov process transitions through sleeping states and listening states periodically, representing the duty cycle operation. In the listening states, the node listens to the channel. Thus, both locally generated packets and relay packets can arrive. In the sleeping states, the node turns off its transceiver and only local packets can arrive. The number of states in $\{I_n\}$ is $L_{\mathrm{c}} = T_{\mathrm{sl}}/T_{\mathrm{u}} + T_{\mathrm{a}}/T_{\mathrm{u}} = T_{\mathrm{p}}/T_{\mathrm{u}}$, where T_{u} is the time unit, and T_{p} is the duration of a duty-cycle period. A large T_{u} can reduce the number of states in the DTMC, thus, reducing computation cost for the model, but at the cost of reducing the granularity and accuracy of the result.

When a packet arrives, the node terminates the quiescent process and begins the first layer of communication process $\{C_n\}$. In each $\{C_n\}$ layer, the node keeps transmitting beacon packets. The number of states in $\{C_n\}$ is $L_{\mathrm{b}} = T_{\mathrm{b}}/T_{\mathrm{u}}$, where T_{b} is the beacon time-out.

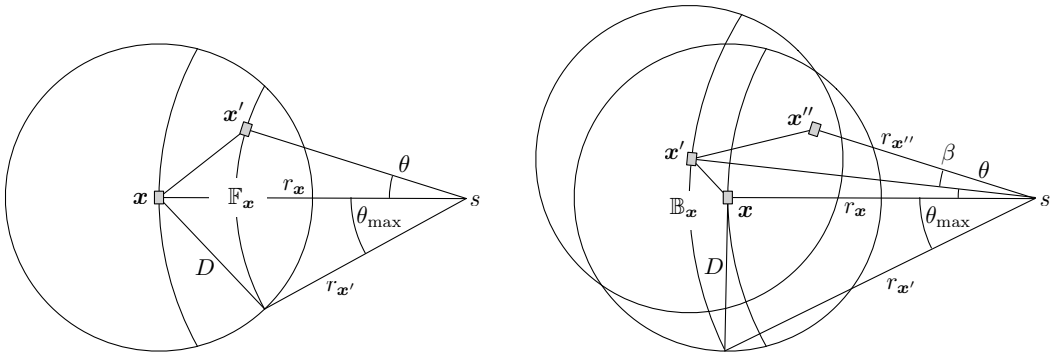
If a node receives RTS responses from other nodes, it starts transmitting the data packet to the first responding node. Retransmissions are conducted in case of a transmission failure. Since only neighbor nodes that receive the beacon packets with a high SNR will response, a high quality wireless channel is guaranteed. Moreover,

in most WSN applications, the traffic rate is low, and the chance of packet collision with other nodes is small. Therefore, data packets transmitted successfully in limited number of (re)transmission attempts, which takes negligible time compared to the sleeping cycle T_{p} (usually longer than 10 s). Thus, $\{C_n\}$ only contains transmission states. When the first RTS packet is received, the transmission terminates in a success. When the beacon transmission times out, the packet is dropped, and the transmission terminates in a failure. In either way, the node enters the lower layer. Note that the beacon timeout T_{b} is usually chosen equal to the cycle T_{p}. This is to ensure that each neighbor node can receive the beacon messages within their duty cycle period. The entire beacon communication process before packet delivery or timeout is regarded as a single transmission attempt. Thus, each communication layer $\{C_n\}$ contains only one block of $\{Z_n\}$.

3.6.2 Constructing the DTMC $\{X_n\}$

The transition probability matrices in $\{I_n\}$ and $\{C_n\}$, are obtained according to the Markov structure in Figure 3.6. In either $\{I_n\}$ or $\{C_n\}$, there is only one initial state (denoted by "begin") with probability of 1. States with outgoing transitions denoted by "success" or "fail" have a probability to complete the current process in a success or failure, respectively. The transition probabilities among states are shown in Figure 3.6. Note that transitions with a probability of 1 are not labeled. The transition probabilities $p_{\mathrm{nr}}(r, v), (1 \leq v \leq L_{\mathrm{b}})$, and the traffic rate $\boldsymbol{\lambda}_I$, $\boldsymbol{\lambda}_C$ are explained in the following.

In the j-th time unit in $\{C_n\}$, a node located at $\boldsymbol{x}$ in ring r has a probability of $p_{\mathrm{nr}}(r, v)$ of not receiving any CTS response, and enters the next state. If in all L_{b} states, the node receives no CTS response, the transmission fails and the packet is



(a) Node $\boldsymbol{y}$ is in the feasible region of $\boldsymbol{x}$.

(b) Node $\boldsymbol{y}$ is in the infeasible region of $\boldsymbol{x}$, and node $\boldsymbol{z}$ is in the feasible region of $\boldsymbol{y}$.

Figure 3.7: The feasible region and infeasible region around node $\boldsymbol{x}$, divided into small areas.

dropped. On the other hand, if in any of the states, a CTS response is received, the node transmits the packet and the transmission succeeds. The probability $p_{\rm nr}(r, v)$ is the conditional probability that given the transmissions in the previous $v-1$ states fails, the transmissions in the v-th state still fails. For simplicity the hidden terminals are ignored. Hidden terminal effects in high density networks can be easily captured by the model as shown in Section 3.5. Therefore,

$$\begin{aligned} p_{\rm nr}(r,1) &= p_{\rm nr}(r, 1 \sim 1) \\ p_{\rm nr}(r,v) &= p_{\rm nr}(r, 1 \sim v)/p_{\rm nr}(r, 1 \sim v-1),\ 2 \le v \le L_{\rm b} \end{aligned} \tag{3.45}$$

where $p_{\rm nr}(r, 1 \sim v)$ is the probability that during all first v states in $\{C_n\}$, beacon transmission fails, since no CTS packet is received in these states. Therefore,

$$p_{\rm nr}(r, 1 \sim v) = \prod_{\boldsymbol{y}=(r_y,\theta_y)\in\mathbb{F}(\boldsymbol{x})} \left(1 - p_{\rm ex}(r_y) p_{\rm ol}(r_y, v) p_{\rm SNR}(\boldsymbol{x}, \boldsymbol{y})\right), \tag{3.46}$$

where each of the small areas at $\boldsymbol{y}$ is located within the transmission range of $\boldsymbol{x}$,

$\mathbb{C}(\boldsymbol{x})$, and is closer to the sink than $\boldsymbol{x}$ (this range is called the *feasible region of* $\boldsymbol{x}$, $\mathbb{F}(\boldsymbol{x})$, as shown in Figure 3.7(a)); r_y is the distance from the small area to the sink; $p_{\text{ex}}(r_y)$ is the probability that there exists a node in each area, and is given by

$$p_{\text{ex}}(r) = \rho \Delta r \Delta \theta r, \tag{3.47}$$

where ρ is the node density. Moreover, $p_{\text{ol}}(r_y, v)$ in 3.46 is the probability that the active period of a node located r_y away from the sink overlaps with the first j beacon transmission time units of the node at $\boldsymbol{x}$; and $p_{\text{SNR}}(\boldsymbol{x}, \boldsymbol{y})$ is the probability that a packet, transmitted from a node at $\boldsymbol{x}$ to a node at $\boldsymbol{y}$, has an SNR higher than some predefined threshold ψ_{th}. It is obtained by (10) in [107].

The probability that the active period of a node at $\boldsymbol{y}$ overlaps with the first v beacon transmission time units of a node at $\boldsymbol{x}$, $p_{\text{ol}}(r_y, v)$, is derived as follows. If node $\boldsymbol{x}$ receives no response in each of the small areas, at least one of the following statements is true: 1) a node does not exist in the area, 2) at least one node exists but they are sleeping during any of the first v slots, and 3) at least one node exists and is awake, but the SNR of the beacon packet they receive is lower than the predefined threshold ψ_{th}. Node $\boldsymbol{y}$ is awake during any of the first v slots means that the first beacon transmission time unit of node $\boldsymbol{x}$ either coincides with any of the awake time units of node $\boldsymbol{y}$ or coincides with the last $v-1$ sleeping units of node $\boldsymbol{y}$. Thus, $p_{\text{ol}}(r_y, v)$ is given by

$$p_{\text{ol}}(r_y, v) = \sum_{k=1}^{L_{\text{w}}} \boldsymbol{\pi}_{W_k}(r_y) + \begin{cases} \sum\limits_{k=L_{\text{sl}}-v+1}^{L_{\text{sl}}} \boldsymbol{\pi}_{\text{s}}(r_y), & 1 \le v < L_{\text{sl}} \\ \sum\limits_{k=1}^{L_{\text{sl}}} \boldsymbol{\pi}_{\text{s}}(r_y), & v \ge L_{\text{sl}} \end{cases} \tag{3.48}$$

where L_{sl} is the number of sleeping time units in $\{I_n\}$, $\boldsymbol{\pi}_{W_k}(r_y)$ and $\boldsymbol{\pi}_{\text{s}}(r_y)$ are the

equilibrium probability that node $\boldsymbol{y}$ is in the k-th awake state or sleeping state in $\{X_n\}$, respectively. $L_{\rm c}$ and $L_{\rm w}$ are the number of total and awake states in $\{I_n\}$, respectively.

Therefore, $p_{\rm nr}(r, 1 \sim v)$ in (3.46) is determined using (3.47) -(3.48), and $p_{\rm nr}(r, v)$ in (3.45) is obtained using (3.46).

Next, the traffic rate at each state, $\boldsymbol{\lambda}_I$ and $\boldsymbol{\lambda}_C$, are discussed. In sleeping states and listening states, the traffic arrival rate is $\lambda_{\rm lc}$ and $\lambda_{\rm re}(r) + \lambda_{\rm lc}$, respectively. In beacon transmission states, since nodes are assumed not to respond to any relay packets, the traffic rate is $\lambda_{\rm lc}$.

Consider the small area $(r : r + \Delta r, \theta : \theta + \Delta\theta)$, where the forwarded traffic arrives from any node $\boldsymbol{y} = (r_y, \theta_y)$ in the *infeasible region* $\mathbb{B}(\boldsymbol{x}) = \mathbb{C}(\boldsymbol{x}) \setminus \mathbb{F}(\boldsymbol{x})$, as shown in Figure 3.7(b). Therefore $\lambda_{\rm re}(r)$ is given by

$$\lambda_{\rm re}(r) = \frac{\sum_{\boldsymbol{y} \in \mathbb{B}(\boldsymbol{x})} p_{\rm ex}(r_y) \lambda_{\rm o}(r_y) p_{\rm fw}(\boldsymbol{y}, \boldsymbol{x})}{p_{\rm ex}(r) \pi_{\rm li}(r)}, \tag{3.49}$$

where $\lambda_{\rm o}(r_y)$ is the output traffic transmitted from $\boldsymbol{y}$. $\pi_{\rm li}(r)$ is the probability that node $\boldsymbol{x}$ is in any listening state, and is the sum of the probabilities corresponding to all listening states in $\boldsymbol{\pi}(r)$. Moreover, $\lambda_{\rm o}(r_y)$ is calculated by

$$\lambda_{\rm o}(r_y) = \boldsymbol{\lambda}(r_y)(\boldsymbol{\pi}(r_y))^{\rm T}(1 - p_{\rm qfull}(r_y) - p_{\rm drop}(r_y)), \tag{3.50}$$

where $p_{\rm fw}(\boldsymbol{y}, \boldsymbol{x})$ is the probability that a node $\boldsymbol{y}$ forwards a packet to node $\boldsymbol{x}$, among all possible forward targets, and $\boldsymbol{\lambda}(r_y)$ is the traffic rate vector for all states in $\{X_n\}$ for $\boldsymbol{y}$. The probability that the packet is dropped due to beacon transmission timeout, $p_{\rm drop}(r_y)$, is easily obtained as $p_{\rm drop}(r_y) = p_{\rm nr}(r, 1 \sim L_{\rm b})$ (see (3.46)). The probability that the queue is full when the packet arrives, $p_{\rm qfull}(r_y)$, is obtained by $p_{\rm qfull}(r_y) =$

$\boldsymbol{\pi}_M(r_y)\boldsymbol{A}_{\rm u}\mathbf{1}$, where $\boldsymbol{\pi}_M(r_y)$ is the probability vector corresponding to the layer M in $\boldsymbol{\pi}(r_y)$, and $\boldsymbol{A}_{\rm u}$ is given by (3.5) for node $\boldsymbol{y}$. In (3.49), $p_{\rm fw}(\boldsymbol{y},\boldsymbol{x})$ is proportional to the probability that node $\boldsymbol{x}$ is *available* when $\boldsymbol{y}$ transmits a beacon, and is normalized on the total probability of availability for all possible nodes. The probability of availability is given by

$$p_{\rm avail}(\boldsymbol{y},\boldsymbol{x}) = p_{\rm ex}(r)p_{\rm wake}(r)p_{\rm SNR}(\boldsymbol{y},\boldsymbol{x}), \tag{3.51}$$

where $p_{\rm wake}(r) = \sum_{j=1}^{L_{\rm w}} \boldsymbol{\pi}_{W_j}(r)$ is the probability that node $\boldsymbol{x}$ is awake, and $\boldsymbol{\pi}_{W_j}(r)$ is the equilibrium probability that node $\boldsymbol{x}$ is in the j-th active state in $\{X_n\}$. Then, $p_{\rm fw}(\boldsymbol{y},\boldsymbol{x})$ in (3.49) is calculated as

$$p_{\rm fw}(\boldsymbol{y},\boldsymbol{x}) = \frac{p_{\rm avail}(\boldsymbol{y},\boldsymbol{x})}{\sum_{\boldsymbol{z}\in\mathbb{F}(\boldsymbol{y})} p_{\rm avail}(\boldsymbol{y},\boldsymbol{z})}, \tag{3.52}$$

where node $\boldsymbol{z}$, with the polar coordinates (r_z,β), can be in any small area in $\mathbb{F}(\boldsymbol{y})$.

Thus, according to (3.49), the traffic rate of node $\boldsymbol{x}$ at each state is determined. Accordingly, $\{I_n\}$ and $\{C_n\}$ are characterized by:

- The (v,v')-th element in $\boldsymbol{P}_I$ and $\boldsymbol{P}_C$ is the transition probability from state v to v' shown in Fig. 3.6.
- The element in $\boldsymbol{\alpha}_I$ and $\boldsymbol{\alpha}_C$ is 1 for states denoted by a "begin" arrow. Other elements are 0's.
- The element in $\boldsymbol{t}_I^{\rm s}$, $\boldsymbol{t}_C^{\rm s}$, and $\boldsymbol{t}_C^{\rm f}$ is set according to the probability attached to the arrows denoted by "success" and "fail", respectively.
- The elements in $\boldsymbol{\lambda}_I$ that correspond to the sleeping states, and the elements in $\boldsymbol{\lambda}_C$ are set to $\lambda_{\rm lc}$. Other elements in $\boldsymbol{\lambda}_I$ are set to $\lambda_{\rm lc}+\lambda_{\rm re}(r)$.

Then, the equilibrium state probability vector, $\boldsymbol{\pi}(r)$, for the DTMC $\{X_n\}$ is obtained for each node $\boldsymbol{x}$. Consequently, the single-hop delay distribution and end-to-end delay distribution for each ring are obtained according to (3.25) and (3.37), respectively.

In the following section, empirical evaluations are used to validate the analytical model for both protocols.

3.7 Analytical Results and Empirical Validations

The end-to-end delay distribution model has been evaluated using MATLAB to determine the single-hop and multi-hop delay distributions for the TinyOS CSMA/CA MAC protocol (Section 3.5) and the anycast protocol (Section 3.6). The computing environment is a PC with a Xeon 5150 CPU working at 2.66GHz and 2G RAM. Moreover, empirical experiments and TOSSIM based simulations have been conducted to validate the results, according to Chapter 2. Each node generates local traffic according to a Poisson distribution with rate $\lambda_{\boldsymbol{lc}}$, and sends the packets to a sink s. Our experiments with the TelosB motes suggest that it requires on the average 1.7 ms to transfer each packet from the MCU to the RF transceiver and 2.0 ms vice versa. The default radio and timing parameters of the experiments are listed in Table 3.1, and the parameters for the channel model are listed in Table 3.2.

In the experiments, the single-hop delay and end-to-end delay are measured as described in Chapter 2. Next, the evaluation results for TinyOS CSMA/CA protocol and the Anycast protocol are presented in Section 3.7.1 and Section 3.7.2, respectively.

Table 3.1: List of radio and timing parameters for TinyOS CSMA/CA protocol.

Group	Notation	Description	Default Value
Radio	l_{p}	data packet size	40 bytes
	R_{b}	channel bit rate	250 kbps
	P_{t}	transmit power	-15 dBm
Timing	T_{u}	time unit	320 μs
	$T_{\mathrm{ibo}}^{\mathrm{max}}$	maximum initial backoff	9.77 ms
	$T_{\mathrm{cbo}}^{\mathrm{max}}$	maximum congestion backoff	2.44 ms

Table 3.2: List of channel-related constants and parameters.

Group	Notation	Description	Default Value
Channel	P_{n}	noise floor	-105 dBm
	$PL(D_0)$	pass loss at reference distance	52.1 dB
	D_0	reference distance	1 m
	η	pass loss exponent	3.3
	σ^{s}	standard deviation of log-normal fading/shadowing	5.5

3.7.1 Experiments for TinyOS CSMA/CA MAC protocol

3.7.1.1 Single-hop Delay Distribution

First, the single-hop delay distribution of the TinyOS CSMA/CA protocol is evaluated according to the derivations in Section 3.5. For the evaluations, a single hop network is considered where the delay distribution is found for a node, while the neighbor nodes also contend for the channel. Three different network configurations are considered for the evaluations.

In the first configuration, a node continuously transmits locally generated packets

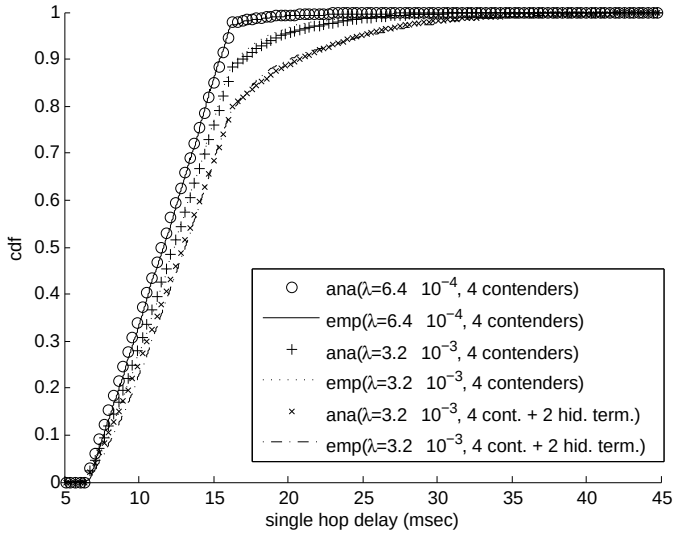
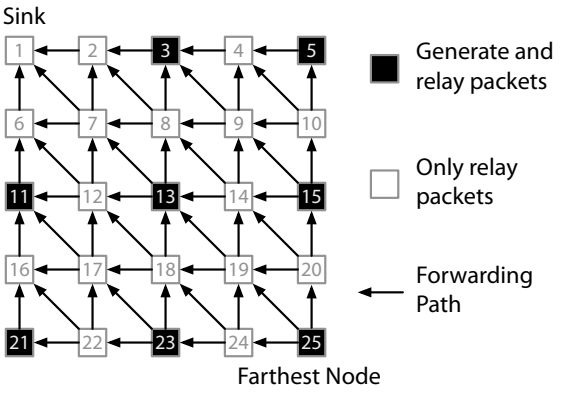


Figure 3.8: The *cdf* of the single hop delay of the CSMA/CA protocol. Both empirical (emp) and analytical (ana) results are shown.

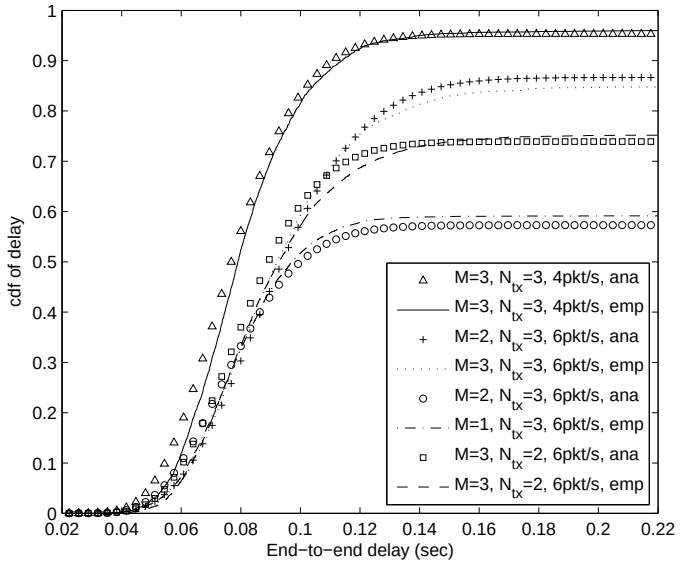
to a receiver node with a data rate of 2 packets per second. This corresponds to $\lambda_{\text{lc}} = 6.4 \times 10^{-4}$ in the analytical model. Four other nodes are used to transmit packets at the same rate to create background traffic for contention. In the second case, the packet rate for all 5 nodes is increased to 10 packets per second. For the third case, two additional nodes with the same packet generation rate are used, but are placed so that they act as hidden terminals for the transmitting node. The single hop delay for $5,000$ packets is recorded for each experiment.

The results of both analytical and empirical validations are shown in Figure 3.8 for the *cdf* of the delay. The results show that a higher traffic rate increases hop delay, which is also captured by our model. In addition, the two hidden nodes introduced in the third case cause heavy contention, and further increase the hop delay. It can be observed that the analytical model accurately captures the effects of hidden nodes. For all cases, the analytical model has less than 2% of error compared to the empirical evaluations.

In our computing environment, the Matlab program runs for less than 10 seconds for a typical scenario with 6 neighbors, with $N_{\text{tx}} = 3$ and $M = 5$ for all nodes.



(a)



(b)

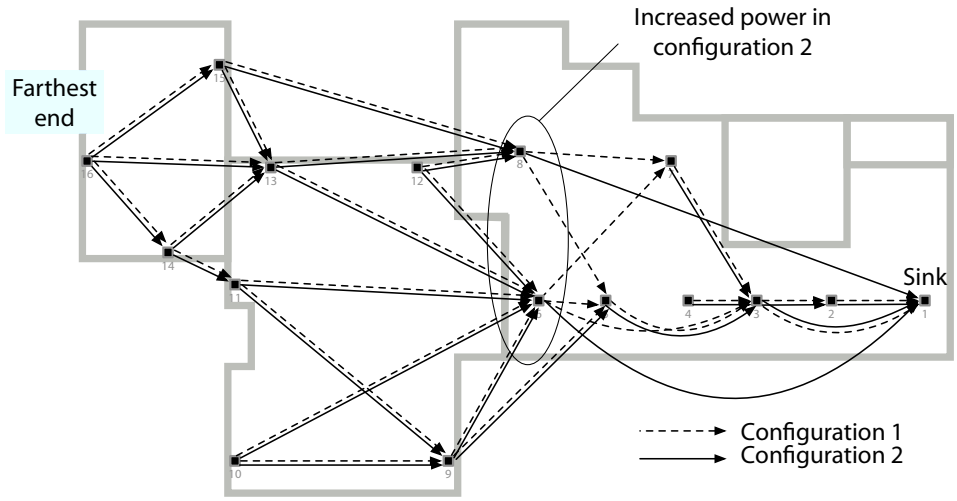
Figure 3.9: (a) The topology and (b) the end-to-end delay distribution for the multi-hop grid experiments with the TinyOS CSMA/CA protocol.

3.7.1.2 End-to-End Delay Distribution

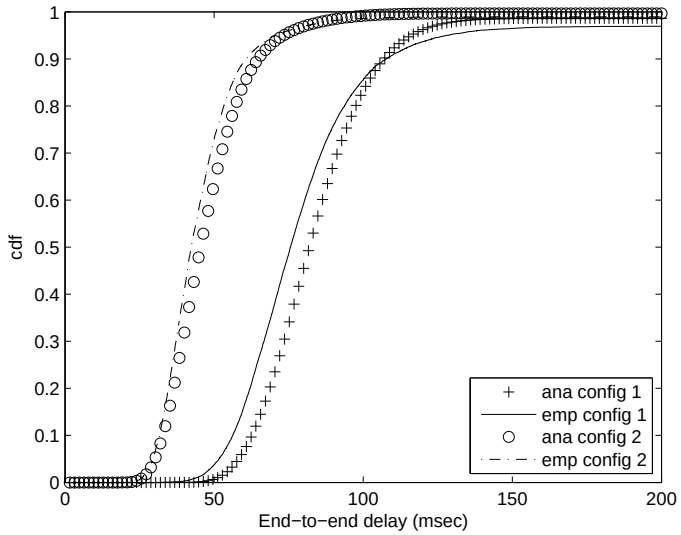
To validate the model for multi-hop networks and illustrate the effects of network parameters in WSNs, two sets of experiments have been performed. First, a network consisting of 25 TelosB nodes are used. The nodes are placed in a 5×5 grid, as illustrated in Figure 3.9(a). Nodes shown as light-colored boxes only relay packets while the 8 dark-colored boxes also generate packets according to a Poisson process. The transmit power for every node is -25 dBm. The generated traffic rate for the 8 nodes, λ_{lc}, the queue length, M, and the maximum number of transmission attempts, N_{tx} are varied to reveal the relationships between each of the parameters and the end-to-end delay distribution. End-to-end delay is measured for approximately $3,000$ packets for each configuration.

The results are shown in Figure 3.9(b). As can be observed, the *cdf* of the analytical model match well with the empirical results with an error less than 5%. The slight difference in these results is partially due to the inaccurate collision models, since the collision range in practice is not an arbitrary area for each node and a transitional area exists around the boundary [107]. The results suggest that heavier traffic leads to a longer end-to-end delay and a lower reliability as can be observed from the asymptotic value of the *cdf*. In addition, by reducing the queue length, M, and the maximum number of transmission attempts, N_{tx}, the reliability decreases. However, when a low delivery rate (e.g., less than 50%) is sufficient, a lower M or N_{tx} does not largely affect the delay performance. More specifically, the average waiting time can be reduced by decreasing the queue capacity and the chance of collisions is decreased since less retransmissions are allowed. This fact is useful when designing applications with nodes having limited memory space.

Experiments are also performed in a realistic indoor environment. A multi-hop



(a)



(b)

Figure 3.10: (a) The topology and (b) the end-to-end delay distribution for the indoor experiments with the TinyOS CSMA/CA protocol.

network of 16 TelosB nodes is located in three rooms as shown in Figure 3.10(a). Two different network configurations are used to illustrated the effects of topology changes. In both configurations, each node generates Poisson traffic of 2 packets per second and the packets are forwarded to the sink as shown in Figure 3.10(a). A

geographical routing protocol is used to determine the forwarding routes based on the distance between each node and the sink. In the first configuration, every node transmits packets with a power of −15 dBm and the routes are shown by dashed lines. In the second configuration, *two nodes* are selected to transmit packets with an increased power of −7 dBm. Therefore, they can directly reach the sink. The routes for the second case are shown in Figure 3.10(a) by solid lines. The *cdf*s of the results are shown in Figure 3.10(b). Accordingly, increasing transmit power in only two nodes significantly impacts the end-to-end delay as also captured by the analytical evaluations.

Table 3.3: List of parameters for the anycast protocol.

Group	Notation	Description	Default Value
Radio	l_{p}	data packet size	40 bytes
	R_{b}	channel bit rate	250 kbps
	P_{t}	transmit power	−15 dBm
	l_{m}	beacon and CTS message size	22 bytes
Timing	T_{p}	duty cycle period	1 s
	T_{a}	active period	0.5 s
	T_{b}	beacon transmission timeout	1 s
	T_{to}	beacon transmission interval	12 ms
	T_{u}	time unit	0.01 s
	$T_{\mathrm{ibo}}^{\mathrm{max}}$	maximum initial backoff	9.77 ms
	$T_{\mathrm{cbo}}^{\mathrm{max}}$	maximum congestion backoff	2.44 ms
Protocol	r_{th}	threshold radius	2.7 m
	ψ_{th}	threshold SNR	10 dBm

3.7.2 Experiments for Anycast Protocol

We first show that the analytical results of the end-to-end delay distribution are validated by the simulation and the testbed experiments. The anycast protocol described in Section 3.6 is implemented in TinyOS 2.0. Our testbed consists of 25 Crossbow TelosB motes. The nodes are randomly placed in a circular area of radius $R = 4.5$ m. Thus the density is roughly $\rho = 0.39$. Each node generates the same amount of local traffic to be sent to the sink according to a Bernoulli process with average rate $\lambda_{\text{lc}} = 0.001$ in each time unit $T_{\text{u}} = 0.01$ s, which equals to 0.1 packet per second. The default duty cycle is $\boldsymbol{x} = 0.5$. The simulation is performed on the same topology. Both the simulations and the testbed experiments have been run for 2.5 hours and the end-to-end delay distribution for a node at distance $r = 4.3$ m is recorded, respectively. Other parameters are shown in Table 3.3.

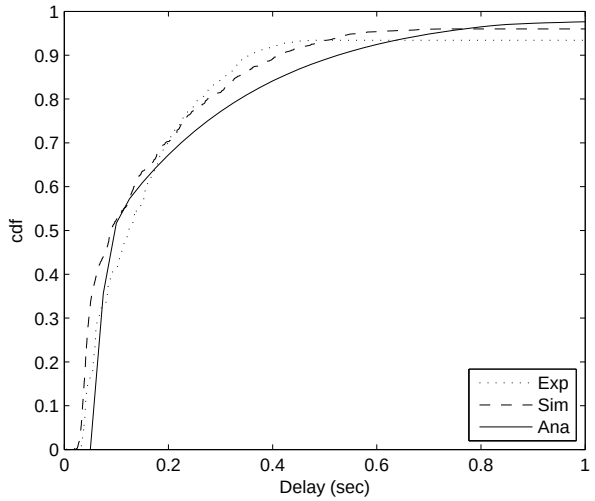


Figure 3.11: The analysis, simulation and experiment results of end-to-end delay distribution with the Anycast protocol for a node with distance $r = 4.3$ m to the sink.

The results are compared with the analytical prediction from the model, as shown in Figure 3.11. It can be observed that the analytical results agree well with both the

simulation result and the testbed experiment result, and the error is less than 10%. Therefore, the simulation is used in the following to validate our model in a larger space and time scale, and for more randomly generated topologies.

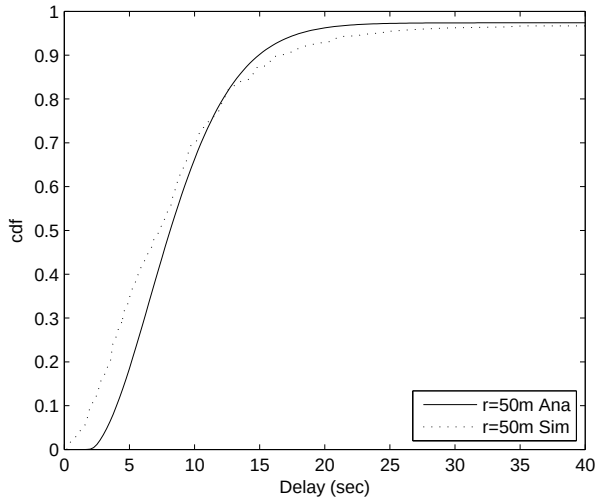
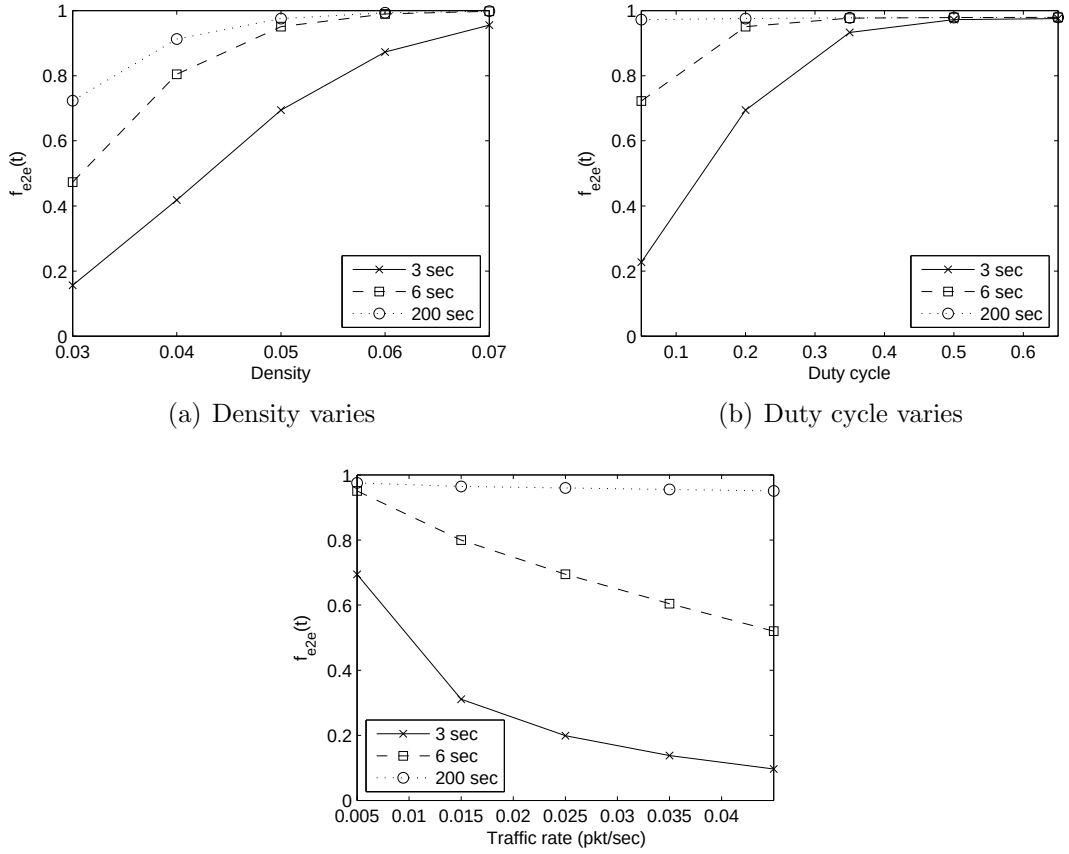


Figure 3.12: The analysis and simulation results of end-to-end delay distribution with the Anycast protocol for a node with distance $r = 50$ m to the sink.

In the second set of evaluations, the network radius is set to 50 m, the transmission power is increased to -10 dBm. Accordingly, the threshold distance is changed to $r_{\text{th}} = 10$ m. Moreover, the network density is $\rho = 0.1$. Durations T_{p}, T_{a}, and T_{b} are 10 sec, 5 sec and 10 sec, respectively, and the traffic rate is 0.01 pkt/sec. Other parameters are left unchanged. 20 different topologies are randomly generated according to a Poisson distribution with the same density. Each topology is simulated for 1 hour. The end-to-end delay distribution from all nodes with a distance of 50 m to the sink are measured. The result is shown in Figure 3.12, along with the analytical results. It can be observed that the analytical result is also within an error of 10% of the simulation result.

Next, using the end-to-end delay distribution modeled in (3.25), we investigate

the relationship between the probability of achieving a given end-to-end delay and various network parameters. In each of the following evaluations, the network density ρ, the duty cycle $\boldsymbol{x}$, and the traffic rate λ_{lc} for all nodes are varied, respectively. The default values for these parameters are 0.02, 0.2, and 0.005 pkt/sec, respectively. Other parameters are kept unchanged from the previous experiment. The network radius is $R = 50$ m.



(a) Density varies

(b) Duty cycle varies

(c) Traffic rate varies

Figure 3.13: The relationship between network parameters and the delivery probability with the Anycast protocol.

The probability that the end-to-end delay of a node at distance $r = 50$ m is smaller than 3 s, 6 s, and 200 s are shown in Figure 3.13. The results in Figure 3.13(a) reveal that when the network density increases, the probability of delivering

packets from the edge to the sink also increases. This is because a network with a higher density tends to have more available relaying nodes at any time. Similarly, as shown in Figure 3.13(b), when the duty cycle increases, nodes have more waking time to relay packets, thus the probability of delivering packets is increased. Finally, Figure 3.13(c) suggests that increasing the traffic rate increases the queueing delay and decreases the probability that nodes are ready to relay packets. Therefore, the probability of delivering packets is smaller as traffic rate increases. It is important to note that given enough time, e.g., 200 s, the delivery probability does not change much when the duty cycle or the traffic rate varies as shown in Figure 3.13(b) and 3.13(c). However, in Figure 3.13(a), the delivery probability after 200 s changes greatly when the network density changes. This is because lower duty cycle and higher traffic rate prolong the packet waiting time. Given enough time, there are still enough nodes to relay the packets. On the other hand, a low network density reduces the number of relaying nodes. Therefore, eventually more packets are lost due to timeout in a low density network.

For any network setup in the experiments above, the calculation for the end-to-end delay distribution during any given duration takes less than 2 minute. On the other hand, the TOSSIM-based simulations determine the delay distribution in the same order of actual time. For example, for a simulated duration of 2 hours, the simulation takes roughly 30 mins. Thus, our analytical approach provides insights significantly faster.

3.8 Conclusions

In this chapter, the probabilistic analysis of end-to-end communication delay in WSNs is presented. A Markov process based on the birth-death problem is used to model

the transmission process in a multi-hop network, and the queuing delay and the effects of wireless channel errors are captured by the model. The developed model is validated by extensive testbed experiments through several network configurations and parameters. The results show that the developed framework accurately models the distribution of the end-to-end delay and captures the heterogeneous effects of multi-hop WSNs.

Chapter 4

Event Detection Delay Distribution

In data monitoring applications, events of interest are detected by sensor nodes, and packets are reported to a sink via multi-hop communication. The event detection delay consists of *discovery delay* for individual nodes to sense and detect the event, and the *delivery delay* for the network to relay reports to the sink. When a given number, n, of packets are received by the sink, the event is considered to be detected. Therefore, the probabilistic analysis of event detection delay is different from the end-to-end communication analysis in Chapter 3 in that, both the discovery delay and the delivery delay should be captured. Moreover, these delays should be analyzed for *a group* of packets instead of individual ones.

In this chapter, the distribution of event detection delay is analyzed for WSNs. We first present a brief survey on the related work in Section 4.1. Then, the problems are formally defined in Section 4.2. Consequently, a spatio-temporal fluid model is developed in Section 4.3 to derive the distribution of event detection delay. Motivated by the fact that queue build up in low-rate traffic is negligible, a low-complexity model is also developed in Section 4.4. Extensive testbed and simulation experiments validate both approaches in several network scenarios in Section 4.5. For the scenarios

in which the framework does not yield accurate results, potential reasons are briefly discussed. Finally, we conclude this chapter in Section 4.6.

4.1 Related Work

Characterizing timing performance for traffic flows in WSNs has been investigated in different contexts. Recently, several models have been developed to analyze probabilistic *bounds* on the delay of traffic flows. As an example, the concept of Network Calculus [20] is extended to derive probabilistic bounds for delay through worst case analysis [12, 32]. However, due to the randomness in and the low power nature of the communication links in WSNs, these worst case bounds cannot capture the stochastic characteristics of end-to-end delay. The communication capacity bounds for wireless networks or WSNs without duty cycle operation are investigated in [28, 33, 41, 60, 99]. However, the applicability of these models to WSNs is limited since in WSNs, the wireless channel utilization is often well below the transmission capacity as nodes are constantly forced into a sleeping state to preserve energy.

The existing studies on event detection delay in WSNs are either focused on (1) the event discovery delay, i.e., the delay until the event is detected by an individual node, or (2) the delivery delay in a broadcast network. In [13], assuming a uniform node deployment and a duty cycle based sensing scheme, an analytical model is developed to derive the distribution of the delay until a stationary or mobile physical event is discovered by any node in the network. In [24], events are considered as detected when it is discovered by a node *connected to the sink*. On the other hand, the communication delay for event detection is investigated in [37] for WSNs deployed in a star topology. When an event occurs, multiple sensors in the network discover it immediately, and transmit their report packets to the central controller. The probability distribution

of the delivery delay for the first $n(n > 0)$ report packets is obtained using a hybrid automata model. However, this model cannot be easily employed for large-scale and multi-hop WSNs, where the model becomes intractable. In contrast, we emphasize the delay before the event is *detected by the sink*, which includes the event discovery delay and the event delivery delay. Moreover, by utilizing fluid-based models, the performance of large-scale multi-hop WSNs can be captured.

Fluid-based models have been widely exploited in IP network analysis [53, 59], and have recently been utilized in the analysis of WSNs [27]. Motivated by the fact that the individual packet behavior is less significant when a flow is concerned, the traffic is considered as a continuous flow instead of individual packets. Accordingly, the complexity of the model can be greatly reduced. Furthermore, spatial fluid-based models have also been utilized recently in [17, 92] to model stationary properties, such as traffic rate and energy consumption for large-scale WSNs. These models greatly reduce the complexity of the (otherwise intractable) problem in either temporal or spatial domains. In our analytical framework, we develop a *spatio-temporal fluid model* for the analysis of event detection delay.

4.2 System Model and Problem Definitions

In this section, we first present the system model, including the random network topology model and a description for the network protocol in consideration. Then, the formal definitions of the problems are given.

4.2.1 Network Topology

In a network deployed to monitor a physical event, nodes are considered to be randomly located according to a Poisson point process, where the node density is ρ. A

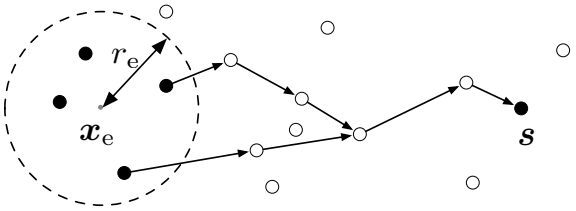


Figure 4.1: The network including the sink and the event generation area.

sink node is deployed at location $\boldsymbol{s} = (x_s, y_s)$, as shown in Figure 4.1.

Assume that at time $t = t_0$, a physical event occurs at location $\boldsymbol{x}_e = (x_e, y_e)$, which is called the event center, and lasts for duration T_e. As shown in Figure 4.1, all sensor nodes within the detection range, r_e, can discover the event. Each sensor node periodically measures the physical world every t_e seconds using its attached sensors. During the event duration $[t_0, t_0+T_e)$, whenever the value of the measurement satisfies a predefined rule, e.g., temperature higher than a given threshold, a report packet of size L is generated and is forwarded to the sink. Each sensor node is assumed to have the same sampling rate, but with a random phase shift, i.e., samples are taken unsynchronized among nodes. Therefore, there is a *discovery delay* between when the event occurs and when it is captured by individual nodes. Due to inherent noise in the sensor readings, n ($n \geq 1$) readings from multiple sensor nodes are required at the sink to successfully detect the event occurrence. Accordingly, we define the following:

Definition 1. *An event is* **n-*detected*** *if n report packets for that event are received by the sink.*

Moreover, each node is implemented with a packet queue of maximum size, M.

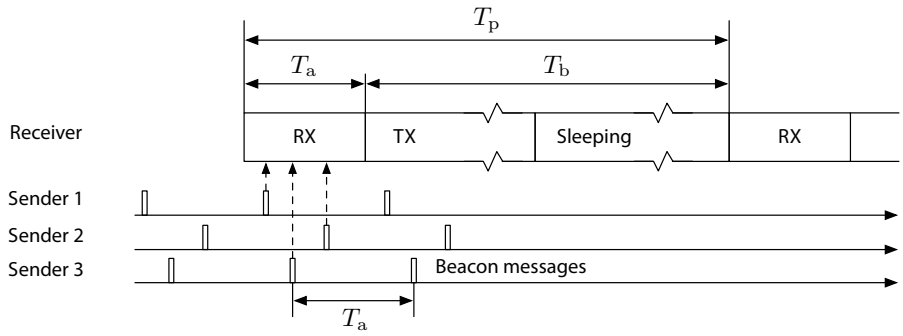


Figure 4.2: The timing of node operations for the anycast protocol studied for event detection delay.

4.2.2 The Anycast Protocol

The delay characteristics of a WSN is dependent on the MAC and routing protocols used in the network. In this chapter, we consider the Anycast protocol described in Chapter 2 for our analysis. A slight change has been made to the timing of the protocol to make the problem tractable, as explained below.

The anycast protocol operation is depicted in Figure 4.2. Each node, except the sink, operates in a duty cycle with a duration T_p. Each cycle is divided into two phases. During the first phase, the listening phase, nodes listen to the channel for any possible incoming traffic. The second phase is the transmission and sleeping phase, in which nodes first try to transmit every packet in the queue. After all packets are transmitted, they turn off radio transceivers to save energy. The duration of these two phases are denoted by T_a and T_b, respectively. The duty cycle, ξ, is defined as $\xi = T_a/T_p$. To obtain a long network lifetime, it is desirable to have a very low duty cycle and hence, generally, $T_a << T_p$. Nodes are assumed *not* to be synchronized.

The packet transmission follows the process described in Chapter 2. When a node has a packet to send, it first broadcasts short beacon messages periodically. If any other node closer to the sink receives the beacon message with a higher signal to noise ratio (SNR) than a given threshold, ψ_{th}, it sends back a CTS message. The first

node that responds with a CTS message is chosen as the next-hop node. Finally, the sender transmits the data packet to it.

The transmission interval of beacon messages is set equal to T_{a}, as shown in Figure 4.2, to ensure that other nodes can receive the beacon messages when they are listening. Therefore, during the listening phase, each node receives a beacon message from its neighbors, if they transmit beacon messages. We assume that all messages from different nodes do not collide with each other. This is a valid assumption, because the duty cycle is usually very small in monitoring applications. Moreover, the beacon and CTS messages are very short and are unlikely to collide. Although data packets may be longer, their length is still usually very small compared to the listening period. In the rare event where data packets collide, their senders can utilize retransmissions after a short amount of delay to ensure delivery. For example, in a typical monitoring WSN application, the operation cycle T_{p} may be set to 10 s, and the listening phase duration T_{a} may be set to 100ms to achieve a 1% duty cycle, as shown in Figure 4.2. The transmission duration of a beacon message or a CTS message is usually less than 1ms, and the transmission duration for a data packet with 40 bytes is less than 2ms for many WSN platforms such as MicaZ and TelosB. The collision probability in this case is minimal and can be neglected. The testbed evaluations reveal that these assumptions are reasonable as discussed in Section 4.5.

Note that during a listening phase, a node can only receive a single beacon message from any other node. Thus, at most one packet can be transmitted from node $\boldsymbol{x}$ to node $\boldsymbol{y}$ for any neighbor nodes $(\boldsymbol{x}, \boldsymbol{y})$ during a duty cycle T_{p}. However, a node may receive multiple beacon messages and respond to them during a listening phase. Thus, it is possible for a node to receive multiple data packets in each listening phase from multiple senders. In such scenario, all packets received are stored in the queue. New packets are dropped if they arrive when the queue is full.

Packets are transmitted in a FIFO basis, until buffered packets are all transmitted. A node may transmit multiple packets to multiple neighbors, but can only transmit one packet to each single neighbor in each duty cycle. After all packets are transmitted, the node turns off its transceiver to save energy, until the next listening phase starts. If at the end of the transmission/sleeping phase, there are still packets not transmitted, the node stops broadcasting beacon messages and begins listening. The beacon message for the current packet will be resumed in the next transmission/sleeping phase.

4.2.3 Problem Definitions

As explained in Chapter 1, several random factors in the topology and node operation affect the communication in the network. Accordingly, the paths from detecting sensors to the sink are dynamically generated and can be considered random. Thus, the delay characteristics of event detection is modeled based on the following definitions.

Definition 2. *The* **n-*delay*** *of an event is the delay between when the physical event occurs and when the event is n-detected.*

Definition 3. *The* **$(\mathbf{p}, \mathbf{n})$-*delay bound*** *of an event is delay within which the event is n-detected with probability p.*

It is assumed that no in-network processing, such as aggregation, is utilized in the network and their effects are left as a future work. To evaluate the delay characteristics of event detection in WSNs, given network and protocol parameters, n, and p; we are interested in the following problems:

- What is the n-delay distribution of an event?
- What is the average n-delay of an event?

- What is the (p, n)-delay bound of an event?

In Section 4.3 and Section 4.4, the proposed spatio-temporal fluid models are presented to address these questions.

4.3 Transient Analysis of Event Detection

In this section, the spatio-temporal fluid model is presented. The network is represented by a continuous fluid entity distributed in the entire network area. Furthermore, the traffic is not considered as individual packets, but a continuous packet fluid. By utilizing a spatio-temporal fluid model, the complexity of the problem in both spatial and temporal domains is reduced, and becomes tractable. The testbed and simulation evaluations (Section 4.5) reveal that the fluid approximation accurately models the delay characteristics.

Consider a location in the network area denoted by $\boldsymbol{x} = (x, y)$. The fluid network model regards the nodes as a fluid entity over the entire space. Then, in an infinitesimal area around location $\boldsymbol{x}$ with size $\mathrm{d}\boldsymbol{x}$ [1], the amount of nodes is $\rho \mathrm{d}\boldsymbol{x}$, where ρ is the node density. We also denote the *feasible region* of $\boldsymbol{x}$ (the region in the transmission range of $\boldsymbol{x}$ and is closer to the sink) as $\mathbb{F}_{\boldsymbol{x}}$, and the *backward region* of $\boldsymbol{x}$ (the region in the transmission range of $\boldsymbol{x}$ and is farther to the sink) as $\mathbb{B}_{\boldsymbol{x}}$. To describe the fluid traffic in the spatial fluid network, the following traffic concepts are introduced:

Definition 4. *The generated, incoming, and outgoing traffic rate density for an infinitesimal area* $\mathrm{d}\boldsymbol{x}$ *is respectively defined as the average number of packets generated, received, and transmitted by the nodes within the area, if any, in an infinitesimal duration* $\mathrm{d}t$*, divided by the duration* $\mathrm{d}t$*, and the size of the area* $\mathrm{d}\boldsymbol{x}$*.*

[1] With a slight abuse of denotation, this infinitesimal area is henceforth denoted by $\mathrm{d}\boldsymbol{x}$.

In other words, the traffic rate densities define the speed at which packets are generated, received, and transmitted in unit space, respectively. In the transient analysis, their values change over time, and thus, are functions of t. The generated, incoming, and outgoing traffic rate density are denoted by $g_{\boldsymbol{x}}(t)$, $\lambda_{\boldsymbol{x}}(t)$, and $\omega_{\boldsymbol{x}}(t)$, respectively. Note that by assuming a fluid model, the *amount of nodes* in an infinitesimal area $\mathrm{d}\boldsymbol{x}$, and the *amount of packets* sent in an infinitesimal duration $\mathrm{d}t$, are not necessarily an integer number.

Definition 5. *The buffered traffic density for an infinitesimal area* $\mathrm{d}\boldsymbol{x}$ *is defined as the average number of packets buffered in the queue by the nodes within the area divided by the size of the area* $\mathrm{d}\boldsymbol{x}$.

The buffered traffic density is also a function of t, and is denoted by $q_{\boldsymbol{x}}(t)$.

In the following, we derive the set of equations that describe the fluid traffic characteristics of the network after $t = t_0$. Without loss of generality, let $t_0 = 0$. For each node, the generated traffic rate density is given by

$$g_{\boldsymbol{x}}(t) = \begin{cases} \dfrac{\rho}{t_{\mathrm{e}}}, & |\boldsymbol{x} - \boldsymbol{x}_{\mathrm{e}}| < r_{\mathrm{e}}, \text{and } 0 \leq t < T_{\mathrm{e}}, \\ 0, & \text{otherwise}, \end{cases} \tag{4.1}$$

where ρ is the density, t_{e} is the reporting interval, and $|\boldsymbol{x} - \boldsymbol{x}_{\mathrm{e}}|$ denotes the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}_{\mathrm{e}}$. During an infinitesimal duration $\mathrm{d}t$, the amount of arriving traffic, along with the traffic already stored in the queue is

$$a_{\boldsymbol{x}}(t) = q_{\boldsymbol{x}}(t) + (\lambda_{\boldsymbol{x}}(t) + g_{\boldsymbol{x}}(t)) \cdot \mathrm{d}t. \tag{4.2}$$

which is the available traffic that needs to be transmitted.

For each infinitesimal area $\mathrm{d}\boldsymbol{y}$ in the feasible forwarding region $\mathbb{F}_{\boldsymbol{x}}$ of $\boldsymbol{x}$, the amount

of nodes with good channel quality is

$$c_{\boldsymbol{x},\boldsymbol{y}} = \rho \cdot p_{\boldsymbol{x},\boldsymbol{y}}(\psi_{\text{th}}), \tag{4.3}$$

where $p_{\boldsymbol{x},\boldsymbol{y}}(\psi_{\text{th}})$ is the probability that the CTS message sent from a node at $\boldsymbol{y}$ has a higher SNR than a given threshold ψ_{th} when received by the node at $\boldsymbol{x}$ ((10) in [107]). Thus, the total amount of nodes in $\mathbb{F}_{\boldsymbol{x}}$ with good channel quality is

$$c_{\mathbb{F}_{\boldsymbol{x}}} = \int_{\mathbb{F}_{\boldsymbol{x}}} c_{\boldsymbol{x},\boldsymbol{y}} \mathrm{d}\boldsymbol{y}. \tag{4.4}$$

Note that between any pair of nodes, at most one packet can be transmitted in a cycle T_{p}. Thus the maximum amount of traffic transmitted during a cycle T_{p} from $\mathrm{d}\boldsymbol{x}$ to anywhere in $\mathbb{F}_{\boldsymbol{x}}$ is

$$\Omega_{\boldsymbol{x}}^{\max} = \rho \mathrm{d}\boldsymbol{x} \cdot c_{\mathbb{F}_{\boldsymbol{x}}}. \tag{4.5}$$

Since the traffic is considered as a fluid and a packet takes one cycle to be transmitted between a pair of nodes, in $\mathrm{d}t$, the maximum amount of traffic sent from $\mathrm{d}\boldsymbol{x}$ is $\Omega_{\boldsymbol{x}}^{\max} \cdot \mathrm{d}t/T_{\text{p}}$. In the case where each node in $\mathrm{d}\boldsymbol{x}$ has less than 1 available packet in its queue, i.e., $a_{\boldsymbol{x}}(t) < 1 \cdot \rho$, it still takes an entire cycle to transmit them. In this case the actual transmitted traffic during $\mathrm{d}t$ is $\frac{a_{\boldsymbol{x}}(t)}{1 \cdot \rho} \cdot \Omega_{\boldsymbol{x}}^{\max} \cdot \frac{\mathrm{d}t}{T_{\text{p}}}$. Accordingly, the transmitted traffic rate density at $\boldsymbol{x}$ is

$$\begin{aligned} \omega_{\boldsymbol{x}}(t) &= \min\left[1, \frac{a_{\boldsymbol{x}}(t)}{1 \cdot \rho}\right] \Omega_{\boldsymbol{x}}^{\max} \frac{\mathrm{d}t}{T_{\text{p}}} \cdot \frac{1}{\mathrm{d}\boldsymbol{x}\mathrm{d}t} \\ &= \min\left[a_{\boldsymbol{x}}(t), \rho\right] \frac{c_{\mathbb{F}_{\boldsymbol{x}}}}{T_{\text{p}}}, \end{aligned} \tag{4.6}$$

where $a_{\boldsymbol{x}}(t)$ is the available traffic density given by (4.2).

The outgoing traffic in each infinitesimal area is equally distributed to every node with good channel quality in its feasible region. Thus, the incoming traffic rate density, $\lambda_{\boldsymbol{x}}(t)$, that is received from each infinitesimal area in the backward region, $\mathbb{B}_{\boldsymbol{x}}$, is given by

$$\lambda_{\boldsymbol{x}}(t) = \int_{\mathbb{B}_{\boldsymbol{x}}} \omega_{\boldsymbol{y}}(t) \cdot \frac{c_{\boldsymbol{y},\boldsymbol{x}}}{c_{\mathbb{F}_{\boldsymbol{y}}}} \mathrm{d}\boldsymbol{y}. \tag{4.7}$$

Within duration $\mathrm{d}t$, the change in buffered traffic density is

$$\mathrm{d}q_{\boldsymbol{x}}(t) = \Big(g_{\boldsymbol{x}}(t) + \lambda_{\boldsymbol{x}}(t) - \omega_{\boldsymbol{x}}(t)\Big)\mathrm{d}t, \tag{4.8}$$

and the buffered traffic density at time $t + \mathrm{d}t$ changes to

$$q_{\boldsymbol{x}}(t + \mathrm{d}t) = q_{\boldsymbol{x}}(t) + \mathrm{d}q_{\boldsymbol{x}}(t) \tag{4.9}$$

Thus, (4.6), (4.7), (4.8) and (4.9) describe the traffic dynamics of the network after $t = t_0$. Given the initial value of $q_{\boldsymbol{x}}(t_0)$, the traffic rates in the network can be evaluated for any time instance $t > t_0$. Accordingly, the total incoming traffic rate at the sink, which models the total number of packets received by the sink, can be obtained. Note that within the transmission range of the sink, the outgoing traffic rate density in (4.6) becomes

$$\omega_{\boldsymbol{x}}(t) = a_{\boldsymbol{x}}(t), \tag{4.10}$$

since the sink is always awake and the traffic can all be transmitted to the sink directly. Moreover, for these nodes, in (4.7), the backward region $\mathbb{B}_{\boldsymbol{x}}$ excludes the areas within the transmission range of the sink. Then, at the sink, the incoming

traffic rate is calculated as

$$\Lambda(t) = \int_{\boldsymbol{x}:|\boldsymbol{x}-s|\leq r_{\text{th}}} \omega_{\boldsymbol{x}}(t)\mathrm{d}\boldsymbol{x}, \tag{4.11}$$

where r_{th} is the distance threshold around the sink within which all nodes directly send packets to the sink.

To calculate the incoming traffic rate at the sink, the entire network area is discretized into small areas, and the time is divided into small time steps. Initially, the buffered traffic density for every infinitesimal area in the network at time $t = 0$ is $q^0_{\boldsymbol{x}}$. $\lambda_{\boldsymbol{x}}(t)$ and $\omega_{\boldsymbol{x}}(t)$ are set as 0. Then, $\omega_{\boldsymbol{x}}(t)$ and $\lambda_{\boldsymbol{x}}(t)$ are calculated using (4.6) and (4.7), respectively. Then, $q_{\boldsymbol{x}}(t)$ is updated for the next time step according to (4.8). This process is repeated for each time step, and $\Lambda(t)$ as a function of t is obtained.

Although the packets are generated with a periodic pattern, the randomness introduced by the routing path and the communication delays results in stochastic behavior for the arrival of packets after multiple hops at the sink, especially in large-scale networks. More specifically, empirical experiments reveal that the traffic arrival process can be approximated by a Poisson process as discussed in Chapter 2. To obtain the n-delay distribution from $\Lambda(t)$, the traffic arrival process is considered a Poisson process with variable rate according to $\Lambda(t)$. The evaluations in Section 4.5 also validate the accuracy of this assumption. Consequently, the n-delay distribution, $f_n(t)$, of the nonhomogeneous Poisson process is given by [34, Ch. 2.4]:

$$f_n(t) = \frac{[\hat{\Lambda}(t)]^{(n-1)}\Lambda(t)\mathrm{e}^{-\hat{\Lambda}(t)}}{(n-1)!}, \tag{4.12}$$

where $\hat{\Lambda}(t)$ is the integral of $\Lambda(t)$ over duration $(0, t]$.

Accordingly, we have the following theorem:

Theorem 2. *For a WSN system described in Section 4.2, the average n-delay and the (p, n)-delay bound of an event are*

$$\mu(n) = \int_0^\infty t f_n(t) \mathrm{d}t, \tag{4.13}$$

$$j(p, n) = f_n^{-1}(p), \tag{4.14}$$

respectively, where $f_n(t)$ is given by (4.12).

Proof. Since $f_n(t)$ in (4.12) is the *pdf* of the n-delay, (4.13) and (4.14) are directly obtained according to the definition of the *pdf*. □

4.4 Simplified Delay Model

The spatio-temporal fluid model presented in Section 4.3 greatly lowers the complexity of the problem. In the model, the entire network area is discretized into small areas, and the traffic rates are calculated for each small area in each time step. To achieve a high accuracy, the size of small areas and the duration of time steps are usually chosen to be very small. In this section, we provide a simplified model to further reduce the calculation complexity. In this simplified model, the network area is divided into small rings. Thus, the spatial calculation complexity is reduced from 2D to 1D.

The simplified model assumes that the traffic is very low in the network, which is typical for many WSN applications. Thus, the queueing effect can be neglected. Moreover, based on the channel-aware next-hop selection explained in Section 4.2, it is assumed that the channel error is negligible within a transmission range of R. For a node located at $\boldsymbol{x}$, after it receives a packet (locally generated or forwarded), in the duration t, the probability that there is no node in its feasible forwarding region $\mathbb{F}_{\boldsymbol{x}}$

waking up is

$$\begin{aligned} p_{\boldsymbol{x}}^{\mathrm{nf}}(t) &\approx \prod_{\boldsymbol{y}=(l,\theta)\in\mathbb{F}_{\boldsymbol{x}}} \left[1-\rho \mathrm{d}\boldsymbol{y} p^{\mathrm{wake}}(t)\right] \\ &= \left[1-\rho \mathrm{d}\boldsymbol{y} p^{\mathrm{wake}}(t)\right]^{\frac{A_{\mathbb{F}_{\boldsymbol{x}}}}{\mathrm{d}\boldsymbol{y}}} \\ &= \exp\left(-A_{\mathbb{F}_{\boldsymbol{x}}}\rho p^{\mathrm{wake}}(t)\right), \end{aligned} \tag{4.15}$$

where the product is conducted over $\mathbb{F}_{\boldsymbol{x}}$, divided according to the polar coordinates originated at $\boldsymbol{x}$, ρ is the network density, $A_{\mathbb{F}_{\boldsymbol{x}}}$ is the size of $\mathbb{F}_{\boldsymbol{x}}$, and $p^{\mathrm{wake}}(t)$ is the probability that a node in each region wakes up during the period t. Since the wake period of each node is unsynchronized with each other, $p^{\mathrm{wake}}(t)$ is irrelevant to the location. Moreover, since each node wakes up at uniformly distributed times, we have

$$p^{\mathrm{wake}}(t) = \begin{cases} \dfrac{t}{T_{\mathrm{p}}}, & 0 \le t \le T_{\mathrm{p}} \\ 1, & t > T_{\mathrm{p}} \end{cases}. \tag{4.16}$$

Therefore, the probability that at least one node in $\mathbb{F}_{\boldsymbol{x}}$ wakes up during t is

$$p_{\boldsymbol{x}}^{\mathrm{fwake}}(t) = 1 - p_{\boldsymbol{x}}^{\mathrm{nf}}(t) = \begin{cases} 1-e^{-A_{\mathbb{F}_{\boldsymbol{x}}}\rho t/T_{\mathrm{p}}}, & 0 \le t \le T_{\mathrm{p}} \\ 1-e^{-A_{\mathbb{F}_{\boldsymbol{x}}}\rho}, & t > T_{\mathrm{p}} \end{cases}. \tag{4.17}$$

This is exactly the *cdf* of the single hop delay. Therefore, the *pdf* of single hop delay for a node at $\boldsymbol{x}$ is

$$f_{\mathrm{sh}(\boldsymbol{x})}(t) = \mathrm{d}p_{\boldsymbol{x}}^{\mathrm{fwake}}(t)/\mathrm{d}t = \begin{cases} \dfrac{A_{\mathbb{F}_{\boldsymbol{x}}}\rho}{T_{\mathrm{p}}} e^{-A_{\mathbb{F}_{\boldsymbol{x}}}\rho t/T_{\mathrm{p}}}, & 0 \le t \le T_{\mathrm{p}} \\ 0, & t > T_{\mathrm{p}} \end{cases}. \tag{4.18}$$

The end-to-end delay distribution from location $\boldsymbol{x}$ to the sink can be found as the

convolution of single-hop delay distributions in the path as explained in Chapter 3. Thus, the *pdf* of end-to-end delay from $\boldsymbol{x}$ to the sink is

$$f_{\mathrm{e2e}(\boldsymbol{x})}(t) = \int_{\boldsymbol{y}\in\mathbb{F}_{\boldsymbol{x}}} f_{\mathrm{e2e}(\boldsymbol{x}')} * f_{\mathrm{sh}(\boldsymbol{x})}(t)\rho \mathrm{d}\boldsymbol{y}, \tag{4.19}$$

where $\boldsymbol{y}$ is the location (l, θ). Note that since the queueing effect is neglected, the nodes with the same distance to the sink have the same end-to-end delay to the sink. Therefore, the end-to-end delay distribution is calculated only once for all nodes with the same distance to the sink. This fact results in a significant reduction on the calculation time.

Suppose the packet generation function for a node at $\boldsymbol{x}$ is $g_{\boldsymbol{x}}(t)$, then the packet reception rate from $\boldsymbol{x}$ by the sink is

$$\lambda_{\boldsymbol{x}}(t) = g_{\boldsymbol{x}} * f_{\mathrm{e2e}(\boldsymbol{x})}(t). \tag{4.20}$$

Then, the packet reception rate at the sink is the sum of traffic generated from each location in the event detection region. Thus,

$$\Lambda(t) = \int_{\boldsymbol{x}\in\mathbb{E}} g_{\boldsymbol{x}} * f_{\mathrm{e2e}(\boldsymbol{x})}(t)\mathrm{d}\boldsymbol{x}, \tag{4.21}$$

where $\mathbb{E}$ is the region within the detection range, r_{e}, of the event location, $\boldsymbol{x}_{\mathrm{e}}$, i.e., $\mathbb{E} = \{\boldsymbol{x} : |\boldsymbol{x} - \boldsymbol{x}_{\mathrm{e}}| \leq r_{\mathrm{e}}\}$.

Finally, the distribution of event detection delay is obtained by using (4.21) in (4.12), and the average n-delay and the (p, n)-delay bound of an event are obtained by Theorem 2.

4.5 Testbed Validation and Simulation Results

To evaluate the accuracy of our proposed analytical framework, testbed experiments and simulations are conducted. The average n-delay and the (p, n)-delay bound of an event in the experiments and simulations are used to compare against the framework. The spatio-temporal fluid model in the framework is implemented using C++ and the simplified model is implemented using MATLAB. In this section, we show that our models provide a high accuracy against both empirical experiments and simulations.

4.5.1 Validation of the Event Detection Delay Analysis

Table 4.1: List of radio, timing and protocol related constants and parameters.

Group	Notation	Description	Default Value
Radio	l_{p}	data packet size	40 bytes
	l_{m}	beacon and CTS message size	22 bytes
	R_{b}	channel bit rate	250 kbps
Timing	T_{p}	duty cycle period	5 s
	T_{a}	wake period	0.1 s
	T_{b}	beacon transmission timeout	10 s
	$T_{\mathrm{ibo}}^{\mathrm{max}}$	maximum initial backoff	9.77 ms
	$T_{\mathrm{cbo}}^{\mathrm{max}}$	maximum congestion backoff	2.44 ms
	T_{tx}	data packet transmission time	1.6 ms
	T_{to}	beacon transmission interval	0.1 s
Protocol	r_{th}	threshold radius	0.6 m
	ψ_{th}	threshold SNR	10 dBm

We first present the results of the testbed experiments. Our testbed, as described in Chapter 2, consists of 40 Crossbow TelosB motes. The nodes are randomly placed

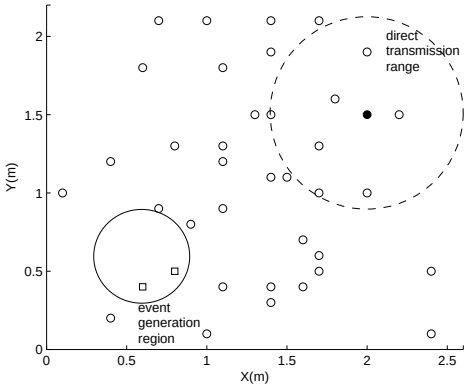
Table 4.2: List of channel-related constants and parameters.

Group	Notation	Description	Default Value
Channel	P_{n}	noise floor	-105 dBm
	$PL(D_0)$	pass loss at reference distance	52.1 dB
	D_0	reference distance	1 m
	η	pass loss exponent	3.3
	σ^{s}	standard deviation of log-normal fading/shadowing	5.5

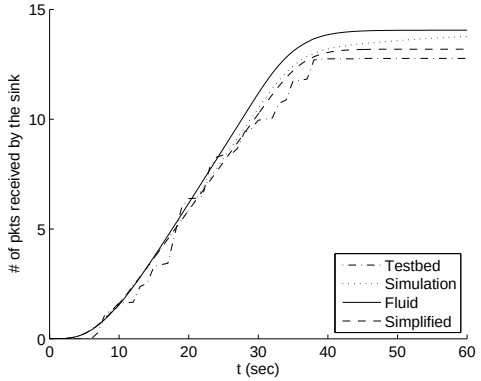
in a rectangular area of size 2×2.4 m^2, as shown in Figure 4.3(a). The node density is thus 7.6 m^{-2}. The sink is located at $(1.5 \text{ m}, 1.9 \text{ m})$ and is marked by a solid dot in Figure 4.3(a). The radio, timing and protocol related parameters are shown in Table 4.1, whereas the channel related parameters (refer to [107] for detailed explanations) are listed in Table 4.2. The transmission power is set to -25 dBm for all nodes.

In the experiment, the event center is located at $(0.5 \text{ m}, 0.5 \text{ m})$. Each node in the network boots up at random time instances. Therefore, they are not synchronized. At time t_0, each of the nodes within r_{e} of the event center (marked as squares in Figure 4.3(a)) starts to generate a series of packets with interval 4 s, and then the generated packets are forwarded to the sink. After 30 s, the nodes stop to generate packets. The experiment is conducted for 62 times. In each test, the delay for all packets received by the sink is logged, and the average total number of packets received as a function of time t after t_0 is depicted in Figure 4.3(b).

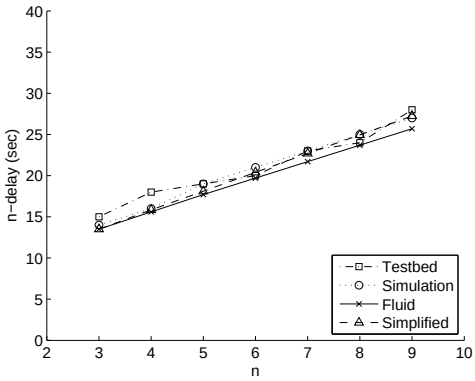
The experiment is also conducted using TOSSIM [56] with the same parameters. Instead of a fixed topology, 100 randomly generated network topologies with the same area and density are used and for each random topology, the simulation is conducted for 5 trials. Then, the total number of packets received at the sink is



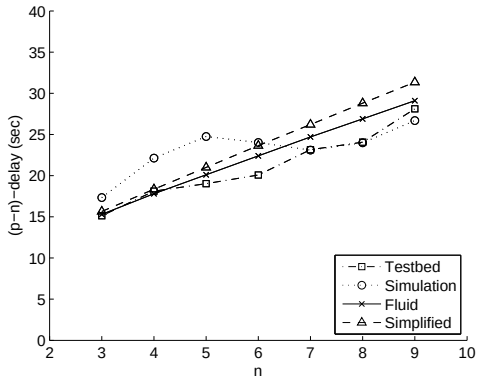
(a) The topology of the testbed experiment.



(b) The reception rate at the sink.



(c) The mean event delay.



(d) The (p, n)-delay bound of the event delay for $p = 0.75$.

Figure 4.3: The map of the testbed experiment, and the results of testbed experiment, the simulation and the models.

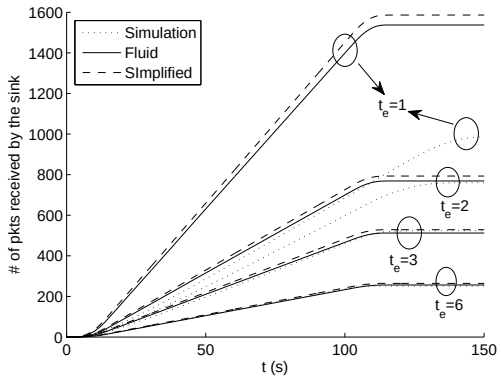
calculated over the trials. The results are shown in Figure 4.3(b) along with the results given by the two analytical models in (4.11) and (4.21). As can be seen in Figure 4.3(b), both testbed experiments and simulations validate the models. To evaluate the accuracy of the models in terms of the average n-delay and (p, n)-delay bound of event detection, from the testbed and simulation result the mean delay for the first $n = 3$ to 9 packets are calculated respectively. The (p, n)-delay bound for $p = 0.75$ is also calculated for different n's. The results are shown in Figure 4.3(c) and Figure 4.3(d). For the majority of the cases, testbed and simulation results are

within 5% of the model. Moreover, the results also confirm the accuracy of TOSSIM simulations, which are used in the following for evaluations of the proposed framework in larger-scale networks.

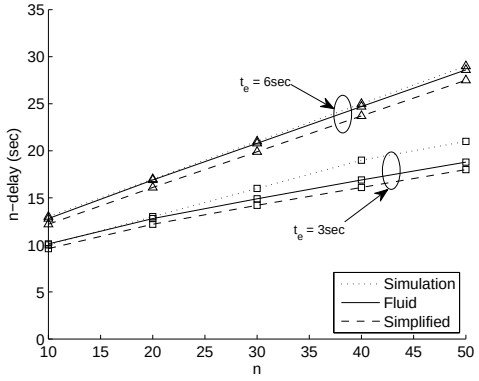
4.5.2 Validation in Larger-Scale Networks

To further evaluate the accuracy of the analytical framework in larger-scale networks, extensive simulations are conducted. Network density, ρ, and the sensing interval, t_{e}, are varied to observe their impact on the event detection delay. Unless otherwise noted, the following parameters are used in the evaluations: The nodes are randomly generated in a square area of size 60×60 m^2, according to a Poisson point process with density $\rho = 0.2$ nodes/m^2. The transmission power is -10 dBm, which corresponds to a transmission range of roughly 10 m. The cycle length is $T_{\mathrm{p}} = 10$ s, in which the listening period is $T_{\mathrm{a}} = 0.1$ s, corresponding to a duty cycle of $\xi = 0.01$. The packet sensing interval is $t_{\mathrm{e}} = 4$ s. The event detection range is $r_{\mathrm{e}} = 5$ m. The sink is located at $(0, 0)$ m, and the event center is located at $(30, 30)$ m. Thus, the distance between the event center and the sink is $d_{es} = 42.4$ m. Other parameters are the same as those in Section 4.5.1.

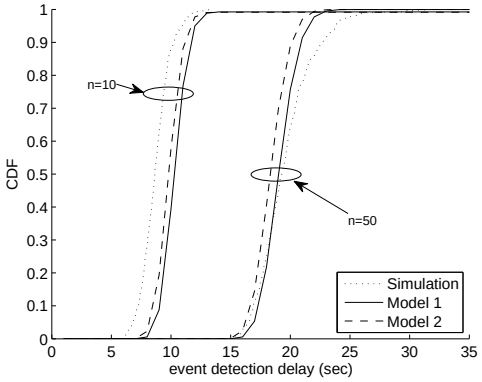
In Figs. 4.4(a)-4.4(d), the effects of sensing interval are shown. In different simulations, the sensing interval t_{e} is set to 1 s, 2 s, 3 s and 6 s respectively for each node, corresponding to packet generation rates of 1, 0.5, 0.333, and 0.167 pkt/s, respectively. The time instances of packet arrivals are logged at the sink, and the average total number of packets received over time is plotted in Figure 4.4(a) along with the model results. It can be observed that for larger sensing intervals, i.e., 3 s and 6 s, both models are accurate with an error less than 5%, whereas for smaller sensing intervals, i.e., 1 s and 2 s, the models accuracy is lower (up to 30% when $n = 10$ and



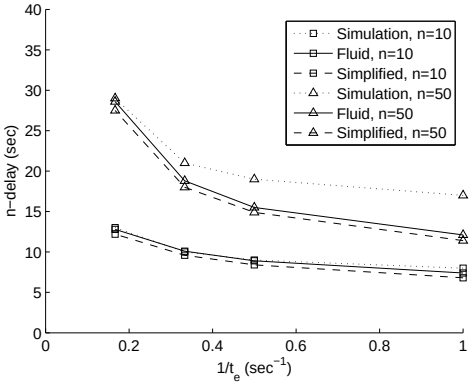
(a) packet reception over time for various sensing intervals t_e.



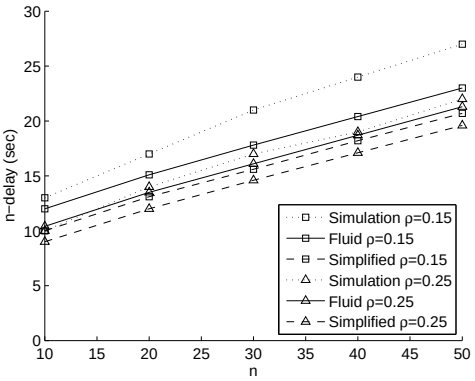
(b) Mean delay vs. n for various sensing intervals t_e.



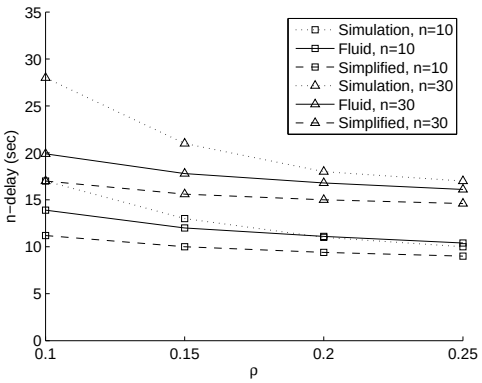
(c) Delay distribution for various sensing intervals t_e.



(d) Mean delay vs. sensing interval for various n.



(e) Mean delay vs. n for various node densities ρ.



(f) Mean delay vs. node density for various n.

Figure 4.4: Mean delay and delay distribution for larger-scale networks. Analytical results are accurate compared to simulations in most cases.

$t_e = 1$ s). This is because with a lower sensing interval, the generated traffic rate is higher. As a result, the received traffic rate at the sink is affected by the clustering effect discussed in Section 4.5.4.

The average n-delay of an event for $n = 10, 20, 30, 40, 50$ packets are shown in Figure 4.4(b) for $t_e = 2, 3$ s respectively. Moreover, Figure 4.4(c) depicts the *cdf* of event detection delay for $n = 10$ and 50 with sensing interval $t_e = 3$ s. The average 10- and 50-delays are also shown in Figure 4.4(d) with varying sensing rates (i.e., the inverse of the sensing interval). In can be observed that the delay reduces when the sensing rate increases, as expected. Note that as Figure 4.4(b) suggests, for $n = 10$, the mean event detection delay for $t_e = 6$ only increases for about 2.5 s from the mean delay for $t_e = 3$ s. This is because for the first 10 packets, the majority of the event detection delay is caused by the packet communication to the sink, no matter how fast packets are generated. In practice, it may be a good idea to set a low sensing rate for sensing nodes if only a few packets are required to detect the event. This saves a great amount of sensing energy with a relatively small tradeoff of event detection delay.

Next, the effects of network density are investigated for values of 0.1, 0.15, 0.2, and 0.25 nodes/m^2. Note that although the network density is changed, the total number of packets generated in the event area remains the same. This is achieved by setting the sensing interval t_e to 2, 3, 4, and 5 s, respectively. By fixing the input traffic, the forwarding capacity of the network with changing density can be analyzed. The mean event detection delay for each density is shown in Figure 4.4(e) as a function of the number of packets n required to detect the event. The 10- and 50-delays of an event are shown in Figure 4.4(f). The figures show that when the network density is lower, the event detection delay slightly increases. This is because when the density is high, each node in the network generally waits for less time before another node

wakes up and becomes available to forward its traffic. Thus, the transmission delay is lower, and the event detection delay is also reduced. However, the traffic reception rate at the sink is still limited by the packet generation rate, which is the same for all cases. Therefore, the average n-delay and (p, n)-delay bound of event detection does not change too much when the density is changed. This suggests in practice, to save sensing energy, the sensing rate of sensor nodes can be reduced. To compensate for the increased event detection delay, additional nodes can be deployed to increase the density.

Note that when the density is as low as 0.15 nodes/m^2, the average n-delay and (p, n)-delay bound from the simulation are higher than the model predictions. This is because when the network density is high, nodes operate well below the forwarding capability. On the other hand, when the density is lower, the network supports less amount of traffic forwarded to the sink. Thus, the detection delay is limited by the lower capability. The error of model prediction is due to the clustering effect discussed in Section 4.5.4.

4.5.3 Comparison Between the Models

In this section, we briefly present the difference between the two proposed models. By assuming that the nodes with the same distance to the sink have the same end-to-end delay to the sink, the simplified model requires $O(\sqrt{A})$ time, where A is the area of the network. On the other hand, since the spatio-temporal fluid model calculates the traffic rates for each location in the entire 2D network, it requires $O(A)$ time. The simplified model significantly outperforms the fluid model in terms of calculation efficiency. Note that both models yield the result in a significantly less time than simulations. For a typical network of 400 nodes, the simulation takes more than one

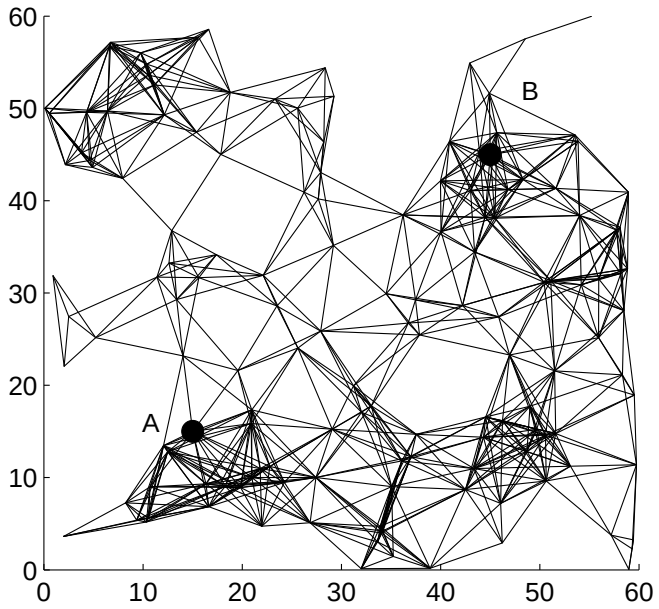


Figure 4.5: Bottlenecks in random WSNs

day to complete, while the complete fluid model takes around 10 minutes to calculate, and the simplified only takes less than 1 minute.

On the other hand, the simplified model becomes inaccurate when the nodes with the same distance to the sink do not have the same end-to-end delay. An example is a non-regular network where nodes density varies over the space. The complete fluid model, however, can be extended to provide accurate result in such networks when the density ρ in (4.1) - (4.6) by $\rho(\boldsymbol{x})$, a density function of corresponding location $\boldsymbol{x}$.

4.5.4 Limitations of the Models

It is shown in Figures 4.3 and 4.4 that both proposed models yield accurate results. The only cases where the result is less accurate is when the density is low, or when the traffic rate is high ($\rho \leq 0.15$ node/m^2 in 4.4(e) or $t_e \leq 2$ s in 4.4(d)). Although such scenarios are generally not typical in WSN applications, it is important to point

out that this is because by considering the nodes as a uniformly distributed fluid over the space, the bottlenecks of random network is neglected. In Figure 4.5 an intuitive view is provided on how the bottlenecks are formed in a random network. As can be seen in Figure 4.5, the nodes form multiple clusters within which the nodes have a high degree of connection and less degree among clusters. This suggests that the transport capacity between A and B, which is usually dependent on the minimum cutset between the two nodes, is limited by the few paths across the clusters. These paths form the bottlenecks of communication [25]. Since the spatio-temporal fluid model assumes a uniform node distribution, it does not capture the bottlenecks, which may cause a higher detection delay when the network density is low, or when the traffic rate is high.

Note that although communication capacity bounds for wireless network communications without duty cycle operations are investigated in [28, 33, 41, 60], as far as we know, the exact solution for communication capacity in random WSNs with duty cycle operations is still an open research issue.

4.6 Conclusions

In this chapter, an analytical framework is proposed to model the event detection in WSNs. In the framework, a spatio-temporal fluid model is utilized to obtain the distribution of the event detection delay. The average delay and soft delay bounds are then obtained. To reduce the calculation complexity, a simplified model is also proposed, motivated by the fact that the queue build up in WSNs is negligible. Testbed experiments and simulations are used to validate the accuracy of both approaches.

Chapter 5

Network Lifetime Distribution

In this chapter, the analysis for the probabilistic network lifetime in WSNs is provided. As the foundation for the lifetime analysis, the probabilistic energy consumption analysis is also presented. The analysis in this chapter is based on the models presented in Chapter 3, since the analysis of both the end-to-end delay and the network lifetime utilize the Discrete-Time Markov model at the node level.

In the following, first, the developed Markov process-based model to analyze the distributions of energy consumption in WSNs is presented, and the distributions of *node lifetime* and *network lifetime* are derived using the energy consumption distribution. Then, it is shown that when the given period is large enough, energy consumption *converges to a Normal distribution*. This result greatly reduces the computation cost for the analysis. Afterward, a case study for the Anycast protocol is provided, and the relationships between network parameters and the lifetime distribution are investigated. Finally, the analysis is validated by realistic testbed experiments and extensive simulations.

5.1 Related Work

The majority of existing work on energy and lifetime analysis in WSNs is focused on the average measures. Average energy efficiency is evaluated for specific protocols [11, 78, 100], and average energy consumption models are proposed in analytical studies [21, 65]. In [21], an analytical energy model is provided to estimate the energy consumption, assuming SMAC [100] and Directed Diffusion [46] as the MAC and routing protocols. The energy consumption of the network is analyzed for low power listening (LPL) operations, and the wasted energy on collision and overhearing are taken into account. In [65], another analytical model is developed to capture the average energy consumption of both software and hardware, and focuses on preamble sampling-based MAC protocols. Stochastic characteristics of energy consumption, however, are not captured by these models.

The effects of routing strategies on energy consumption have also been investigated recently. In [42], a realistic radio model is adopted to analyze the tradeoff between single-hop long-distance transmissions and multi-hop short-distance transmissions. The same problem is also investigated in [97], using an energy model focused on circuit level hardware. These models, while providing a detailed estimation of average energy consumption for certain protocols, do not offer higher-order statistics of energy consumption for generic MAC or routing protocols.

For single node and network lifetime analysis, most of the existing work only investigates average measures. An analytical model is provided in [26] to study the energy consumption and lifetime of two-tier cluster WSNs. Lifetime is measured in terms of data collection "rounds" in which an average amount of energy is consumed for each node. In [54], the lifetime is analyzed for an always-on network, where the energy management problem is most severe. In [48], a detailed energy and lifetime

model is proposed for trigger-driven and duty-cycle driven applications in WSNs. For a given event arrival rate, the model derives the average lifetime of a node. In all of these studies, only the first-order lifetime statistics are investigated.

Probably the most closely related work is the probabilistic lifetime analysis provided in [73], where a cluster-based network topology is considered for event-driven applications. The node lifetime distribution is modeled for a cluster-based network topology, using a TDMA MAC protocol. However, the applications of this analysis to other network topologies, such as mesh topologies and ad hoc networks, and other protocols, such as CSMA-based protocols, have not been shown.

In the following, we present the probabilistic energy consumption and network lifetime analysis by defining the problems first.

5.2 Problem Definition and System Model

In WSNs, energy is consumed by each node for various activities including sensing, data processing, and communication. We assume that each node is equipped with K sensors, and each sensor $k \in \{1, \ldots, K\}$ is used to sense the physical environment every $T_{\mathrm{s},k}$ seconds (subscript "s" refers to "sensing") with an energy consumption of $\varepsilon_{\mathrm{s},k}$. Based on the application requirements, a packet is generated locally if the sensed information satisfies event definitions. For each received and locally generated packet, the node processes the data with an energy consumption of ε_{p}. Moreover, the energy consumption for the communication, ε_{c}, is a variable dependent on the network parameters and the protocols running on each node.

We consider two types of network deployments, the random deployment and the deterministic deployment, as explained in Section 2.1.1. In both cases, each node $\boldsymbol{x}$ is characterized by its input traffic rate, $\lambda(\boldsymbol{x})$, queue length, $M(\boldsymbol{x})$, and battery capacity,

$C(\boldsymbol{x})$. Moreover, the wireless channel between each node is modeled according to a log-normal fading channel model [107], as explained in Section 2.1.2, whereas other channel models can also be used.

For a given network topology and node parameters as described above, we are interested in the following problems:

1. Given a period of time T, what is the energy consumption distribution, $F_{E(\boldsymbol{x},T)}(e)$, of a node at $\boldsymbol{x}$?

2. Given the energy consumption distribution, what is the lifetime distribution, $F_{L(\boldsymbol{x},C(\boldsymbol{x}))}(t)$, of a node at $\boldsymbol{x}$?

3. Given the energy consumption distribution for each node $\boldsymbol{x}$ in the network, what is the distribution of the network lifetime, $F_{NL}(t)$?

These random variables depend on the protocol operation and network topology. In this section, an overview of our solutions for the above problems is provided. The details of the framework are elaborated in Sections 5.3 and 5.4.

5.2.1 Single Node Energy Consumption Distribution

The randomness in energy consumption and the associated lifetime is due to two main components: First, the communication protocol operation induces randomness because of the wireless channel errors and queueing operation. Second, the variations in the network topology results in different nodes consuming different amounts of energy in the network. In the following, we first present the energy consumption distribution for random deployment, which models both cases. Then, a special case for deterministic deployment, e.g., grid topology, is presented, where the randomness due to topology can be ignored.

5.2.1.1 Random Deployment

For a randomly deployed network, the randomness in energy consumption due to topology is caused by the variations in local density. This is captured by considering the randomness due to protocol operation and topology separately. Accordingly, for a node at $\boldsymbol{x}$, the r.v. for energy consumption during a given time period, T, is expressed as the sum of 3 independent r.vs:

$$E(\boldsymbol{x}, T) = E_{\mathrm{s}}(\boldsymbol{x}, T) + E_{\mathrm{cp}}(\boldsymbol{x}, T) + E_{\mathrm{tc}}(\boldsymbol{x}, T), \tag{5.1}$$

where $E_{\mathrm{s}}(\boldsymbol{x}, T)$ is the r.v. of energy consumption for sensing, and $E_{\mathrm{cp}}(\boldsymbol{x}, T)$ is the r.v. of energy consumption for communication and processing. These two terms capture the randomness due to protocol operation by considering a homogeneous density in the network. The last term in (5.1), $E_{\mathrm{tc}}(\boldsymbol{x}, T)$, is an empirically determined zero-mean r.v. that captures the randomness in energy consumption due to topology. Accordingly, the *pdf* of the total energy consumption of a node at $\boldsymbol{x}$ is

$$f_{E(\boldsymbol{x},T)}(e) = f_{E_{\mathrm{s}}(\boldsymbol{x},T)} * f_{E_{\mathrm{cp}}(\boldsymbol{x},T)} * f_{E_{\mathrm{tc}}}(\boldsymbol{x}, e), \tag{5.2}$$

where the *pdf* of the corresponding r.vs. in (5.1) are convolved. Testbed and simulation results are used in Section 5.6 to show that the assumption of independence for randomness due to protocol and topology is accurate.

5.2.1.2 Deterministic Deployment

A large class of WSN applications relies on deterministic deployment of sensor nodes, e.g., grid deployment. This is a special case, where the effects of randomness due to

topology are not observed. Accordingly, (5.2) can be simplified to

$$f_{E(\boldsymbol{x},T)}(e) = f_{E_{\mathrm{s}}(\boldsymbol{x},T)} * f_{E_{\mathrm{cp}}(\boldsymbol{x},T)}(e). \tag{5.3}$$

Next, the distribution of energy consumption for sensing, $f_{E_{\mathrm{s}}(T)}(e)$, is described[1] and the Markov-chain formulation for deriving the distribution of energy consumption for communication and processing, $f_{E_{\mathrm{cp}}(\boldsymbol{x},T)}(e)$, is summarized.

5.2.1.3 Energy Consumption for Sensing

During any given time duration T starting at t_0, i.e., the period $[t_0, t_0+T)$, for some sensor k with periodic sensing interval $T_{\mathrm{s},k}$ and energy consumption per sensing $\varepsilon_{\mathrm{s},k}$, denote the first sensing activity for sensor k after t_0 occurs at t_{k1}. The number of sensing activities during T is then $n_k(T) = \lceil (t_0+T-t_{k1})/T_{\mathrm{s},k} \rceil$. Since t_0 is independent of sensing activities, $t_{k1}-t_0$ is a r.v. uniformly distributed in range $[0, T_{\mathrm{s},k})$. Therefore, the *pmf* of $n_k(T)$ is given by

$$f_{n_k(T)}(n) = \begin{cases} N_{\mathrm{s},k} - n + 1, & n = \lfloor N_{\mathrm{s},k} \rfloor + 1 \\ n + 1 - N_{\mathrm{s},k}, & n = \lfloor N_{\mathrm{s},k} \rfloor \\ 0, & \text{otherwise} \end{cases}, \tag{5.4}$$

where $N_{\mathrm{s},k} = \dfrac{T}{T_{\mathrm{s},k}}$. The *pdf* of energy consumption for all K sensors during T is obtained as

[1] When there is no ambiguity, the parameter $\boldsymbol{x}$ in any node-specific variables is omitted for clarity.

$$
\begin{aligned}
f_{E_{\mathrm{s}}(T)}(e) &= \sum_{k=1}^{K} \sum_{n} n \cdot \delta(e - f_{n_k(T)}(n)) \\
&= \sum_{k=1}^{K} \big[(N_{\mathrm{s},k} - \lfloor N_{\mathrm{s},k} \rfloor) \cdot \delta \left(e - (\lfloor N_{\mathrm{s},k} \rfloor + 1)\, \varepsilon_{\mathrm{s},k} \right) \\
&\quad + (\lfloor N_{\mathrm{s},k} \rfloor + 1 - N_{\mathrm{s},k}) \cdot \delta \left(e - \lfloor N_{\mathrm{s},k} \rfloor\, \varepsilon_{\mathrm{s},k} \right) \big]. \qquad (5.5)
\end{aligned}
$$

5.2.1.4 Energy Consumption for Communication and Processing

Now, we briefly introduce the model for the analysis of the energy consumption for communication and processing, $E_{\mathrm{cp}}(T)$, and leave the details of the model to Section 5.3.

The energy consumed by communication and data processing at each node in the network is modeled by a discrete-time queueing system with time unit T_{u}, which is characterized by its traffic inter-arrival distribution and service process. More specifically, in each time unit, the traffic inter-arrival is modeled according to a Bernoulli process (refer to Section 2.1.3 for detailed explanation), and a variant of the Discrete Time Markov Process (DTMP) proposed in Section 3.3 is used to model the service behavior.

Similar to the DTMP model in Section 3.3, the DTMP for the energy consumption analysis is represented by a Discrete-Time Markov Chain (DTMC) $\{X_n\}$, which is divided into a *Quiescent layer* and M *Communication layers*, where M is the queue capacity, as explained in Section 3.3. Each state in $\{I_n\}$ and $\{C_n\}_m$ represents the activity that is conducted by the node during each time unit of T_{u}, such as sleeping, transmission, or listening. For example, the duty cycle operations are usually represented as chains of sleeping states and active states in $\{I_n\}$, and the number of states

of each type depends on the duty cycle ξ.

Each state v is also associated with an amount of energy, ε_v, consumed for the corresponding activity during T_{u}. The communication and data processing behaviors of each node are represented by transitions among states in $\{X_n\}$. The detailed explanation of this DTMC is provided in Section 5.3. Based on this DTMC, the *pdf* of the single-node energy consumption for communication and data processing, $E_{\mathrm{cp}}(T)$, is found for any given duration T.

5.2.1.5 Energy Consumption Due to Topology Randomness

The randomness in topology for random deployment introduces variation in energy consumption. This randomness is captured by a zero-mean r.v. $E_{\mathrm{tc}}(T)$. We assume nodes are deployed in a 2-D space according to a Poisson point process (PPP). $E_{\mathrm{tc}}(T)$ is approximated as a Normally distributed variable, and its variance is found by utilizing a semi-empirical approach. Together with the distribution of the energy consumption for sensing, $E_{\mathrm{s}}(T)$, in (5.5), and the distribution of the energy consumption for communication and data processing, $E_{\mathrm{cp}}(T)$, the distribution of total energy consumption $E(T)$ is finally derived according to (5.2).

We will also show that when T is large enough, $E(T)$ asymptotically approaches the Normal distribution. The mean and the variance are given in Section 5.3.4.

5.2.2 Node Lifetime and Network Lifetime Distributions

The lifetime distribution of a node depends on the energy consumption distribution during any given period T, and the total capacity of its battery C. The network lifetime distribution depends on the lifetime distribution for each node, and how the network lifetime is defined. For different applications and network topologies, the

network lifetime can be defined differently [23]. While a complete investigation of network lifetime with various definitions is out of scope in this dissertation, we focus on the lifetime defined as follows.

Definition 6. *The network lifetime is defined as the duration before the battery depletion of* ***the first node****.*

In the following, we first explain the Discrete-Time Markov model based analytical framework, which is used to find the single-node energy consumption distribution. Then, in Section 5.4, the node lifetime distribution and network lifetime distribution are found based on the single-node energy consumption distribution.

5.3 Single Node Energy Consumption Distribution

The energy consumed by communication and data processing for a node is represented by the energy costs associated with transitions among states in Markov chain $\{X_n\}$. In the following, based on the discussion in Section 3.3, the construction of states and transitions in $\{X_n\}$ is discussed.

5.3.1 Structure of Markov chain $\{X_n\}$

According to the MAC protocol employed, the structures of $\{C_n\}_m$ and $\{I_n\}$ are parameterized by the following variables: $\boldsymbol{P}_I$, $\boldsymbol{P}_C$, $\boldsymbol{\alpha}_I$, $\boldsymbol{\alpha}_C$, $\boldsymbol{t}_I^{\mathrm{s}}$, $\boldsymbol{t}_C^{\mathrm{s}}$, $\boldsymbol{t}_C^{\mathrm{f}}$, $\boldsymbol{\lambda}_I$, and $\boldsymbol{\lambda}_C$. The definitions of these variables are given in Chapter 3. Accordingly, the transition probability matrix, $\boldsymbol{Q}_X$, of the entire Markov chain $\{X_n\}$ can be found based on (3.11). Then, the equilibrium state probability vector, $\boldsymbol{\pi}$, for $\{X_n\}$ is calculated by

solving $\boldsymbol{\pi}\boldsymbol{Q}_X = \boldsymbol{\pi}$, and the solution to vector $\boldsymbol{\pi}$ is found according to (3.14) and (3.15) in Section 3.3.

5.3.2 Energy Consumption for Communication and Processing

The difference between the model developed for the energy consumption analysis and the one for delay analysis is, there is an energy cost associated with each state in the DTMC $\{X_n\}$. Suppose at the beginning of a time unit T_{u}, the node is in state v of $\{X_n\}$, and during the time unit, the energy consumption of the node for communication and data processing is ε_v. The value of ε_v is obtained from measurement, or is calculated according to the specifications of the hardware platform. An example will be given in Section 5.5 to show how ε_v is calculated. The *cdf* and the *pdf* of $E_{\mathrm{cp}}(T_{\mathrm{u}})$, the energy consumption during the time unit, are $G_v(e) = u(e - \varepsilon_v)$ and $g_v(e) = \delta(e - \varepsilon_v)$, respectively, where $u(\cdot)$ is the unit step function and $\delta(\cdot)$ is the delta function[2]. We denote

$$H^{(1)}_{v,v'}(e) = G_v(e)q_{v,v'} = \Pr\{E_{\mathrm{cp}}(T_{\mathrm{u}}) \le e \cap v \xrightarrow{1} v'\}, \tag{5.6}$$

$$h^{(1)}_{v,v'}(e) = g_v(e)q_{v,v'} = \mathrm{d}H^{(1)}_{v,v'}(e)/\mathrm{d}e, \tag{5.7}$$

where $v \xrightarrow{1} v'$ represents the event that $\{X_n\}$ transitions from state v to v' in one time unit, and $q_{v,v'}$ is the (v, v')-th element of the transition probability matrix, $\boldsymbol{Q}_X$, in (3.11).

For a given period T, the number of time units of T_{u} is $\hat{T} \sim T/T_{\mathrm{u}}$ (since T_{u} is usually chosen to be very small, T is approximated as an integer multiple of T_{u}).

[2]Although a discrete time Markov process is used for the model, the energy consumption is *continuous*. Thus the *pdf*, as opposed to the *pmf*, is used to characterize the distribution.

After $\hat{T}$ time units ($\hat{T} > 1$), the *cdf* of energy consumption becomes

$$H_{v,v'}^{(\hat{T})}(e) = \Pr\{E_{\rm cp}(T) \le e \cap v \xrightarrow{\hat{T}} v'\} = \int_0^e h_{v,v'}^{(\hat{T})}(\epsilon)\mathrm{d}\epsilon$$
$$= \int_0^e \sum_{v'' \in \mathcal{S}} (h_{v,v''}^{(1)} * h_{v'',v'}^{(\hat{T}-1)})(\epsilon)\mathrm{d}\epsilon \tag{5.8}$$

where $\mathcal{S}$ is the set of all states in $\{X_n\}$. Therefore, if the matrix of $h_{v,v'}^{(\hat{T})}(e)$ is denoted by $\boldsymbol{h}^{(\hat{T})}(e)$, then $\boldsymbol{h}^{(\hat{T})}(e)$ is the $\hat{T}$-fold convolution of $\boldsymbol{h}^{(1)}(e)$.

The energy consumption distribution during T depends on the initial state of the system at the beginning of this period, which is usually randomly chosen. Thus, the initial state probability vector is represented by the equilibrium state probability vector $\boldsymbol{\pi}$. After $\hat{T}$ time units, the *pdf* and the *cdf* of the energy consumption are

$$f_{E_{\rm cp}(T)}(e) = \boldsymbol{\pi}\boldsymbol{h}^{(\hat{T})}(e)\mathbf{1},$$
$$F_{E_{\rm cp}(T)}(e) = \int_0^e f_{E_{\rm cp}(T)}(\epsilon)\mathrm{d}\epsilon, \tag{5.9}$$

respectively, where $\mathbf{1}$ is the appropriately dimensioned column vector containing all 1's.

5.3.3 Energy Consumption Due to Topology Randomness

Our experiments indicate that in WSNs with random deployment, the randomness of the topology introduces variation to the energy consumption. This variation is small when T is short, and increases quadratically with T. It is modeled by a zero-mean Normally distributed r.v. $E_{\rm tc}(T)$. This approximation is accurate, because our experiment in Section 5.6.6 shows that, for parameters of interests, the achievable lifetimes given by the proposed framework have an error less than 3% for single node.

The variance of $E_{\text{tc}}(T)$ is expressed as

$$\sigma^2_{\text{tc}}(T) = cT^2, \tag{5.10}$$

where c is the scaling coefficient function determined by network parameters, such as node density ρ, locally generated traffic rate λ_{lc}, and how the employed protocols are impacted by local density variation. For given network parameters and protocols, c is a constant.

To calculate c for a given set of parameters and protocol, a semi-empirical approach is used. Simulations are conducted to find an expression of c as a function of network parameters, such as ρ and λ_{lc}. The obtained c is then used to derive the energy consumption distribution of nodes with the given protocol.

In Section 5.5, a detailed analysis of (5.10) will be provided for an anycast protocol.

The deterministic deployment, e.g., grid topology, is a special case with no randomness in topology. Thus, the scaling coefficient, c, is zero.

5.3.4 Asymptotic Energy Consumption Distribution

If a QBD process is modeled by a DTMC, and each state in the DTMC is associated with a cost, then the sum of the total cost during a given period T asymptotically approaches the Normal distribution as $T \to \infty$ [66]. Thus, considering the energy consumption, ε_v, at each state v as the cost, the total energy consumption for communication and data processing during T asymptotically approaches the Normal

distribution, whose mean and the variance are given by

$$\lim_{T\to\infty} \mu_{\text{cp}}(T) = \hat{T}\mu_{\text{cp,u}} = \hat{T}\boldsymbol{\pi}\boldsymbol{\varepsilon}, \tag{5.11}$$

$$\lim_{T\to\infty} \sigma^2_{\text{cp}}(T) = \hat{T}\sigma^2_{\text{cp,u}} = \hat{T}\left(\sum_{v\in\mathcal{S}}(\varepsilon_v - \boldsymbol{\pi}\boldsymbol{\varepsilon})^2\pi_v + 2\boldsymbol{\beta}\boldsymbol{\varepsilon}\right), \tag{5.12}$$

respectively, where $\hat{T} = T/T_{\text{u}}$ is the number of time units in T, $\mu_{\text{cp,u}}$, and $\sigma^2_{\text{cp,u}}$ are the mean and variance of E_{cp} during each time unit T_{u}. Moreover, $\boldsymbol{\pi}$ is the equilibrium state probability vector of $\{X_n\}$, $\mathcal{S}$ is the set of states in $\{X_n\}$, and π_v is the equilibrium state probability for state v. Finally, $\boldsymbol{\varepsilon}$ is the vector of ε_v for each state v in $\{X_n\}$, and $\boldsymbol{\beta}$ is an intermediate vector variable which is obtained by solving the following set of equations [66]:

$$\boldsymbol{\beta}(\boldsymbol{Q}_X - \boldsymbol{I}) = -\boldsymbol{\gamma}\boldsymbol{Q}_X, \tag{5.13}$$

$$\boldsymbol{\beta}\mathbf{1} = 0, \tag{5.14}$$

where $\boldsymbol{\gamma}$ is a row vector whose v-th element is $(\varepsilon_v - \boldsymbol{\pi}\boldsymbol{\varepsilon})\pi_v$.

Then, the asymptotic distribution for $E_{\text{s}}(T)$, the energy consumption by sensing, is also derived. For each sensor k, when $T \to \infty$, $\left\lfloor \frac{T}{T_{\text{s},k}} \right\rfloor \approx \frac{T}{T_{\text{s},k}} \approx \left\lfloor \frac{T}{T_{\text{s},k}} \right\rfloor + 1$. Thus, (5.5) becomes

$$f_{E_{\text{s}}(T)}(e) \approx \delta\left(e - \sum_{k=1}^{K} \frac{\varepsilon_k T}{T_{\text{s},k}}\right). \tag{5.15}$$

In other words, the energy consumption is approximately linear to T with a constant coefficient equal to $\sum_{k=1}^{K} \varepsilon_k/T_{\text{s},k}$.

Therefore, the following results are obtained:

Theorem 3. *When $T \to \infty$, the energy consumption of a node during T asymptoti-*

cally approaches the Normal distribution, with the mean and variance linear to T and given by

$$\mu(T) = \hat{T}\left(\mu_{\mathrm{cp,u}} + \sum_{k=1}^{K} \frac{\varepsilon_k T_{\mathrm{u}}}{T_{\mathrm{s},k}}\right), \tag{5.16}$$

$$\sigma^2(T) = \hat{T}\sigma^2_{\mathrm{cp,u}} + cT^2, \tag{5.17}$$

where $\hat{T} = T/T_{\mathrm{u}}$ is the number of time units in T.

Proof. The proof is trivial by combining (5.11), (5.12), (5.15), and (5.10). □

5.4 Lifetime Distribution Analysis

Using the *pdf* of energy consumption $f_{E(T)}(e)$ in (5.9) for any given period T, the lifetime distribution of a node, and further, the entire network, can be found as follows.

5.4.1 Single-Node Lifetime Distribution

The r.v. of lifetime for a given node, $L(C)$, is a function of its total battery capacity C. Initially, the node has a battery residual of C. After duration T, the *pdf* of remaining energy in the battery is $f_{C-E(T)}(e)$. The probability that the node has a shorter lifetime than duration T is the probability that the remaining energy after T is lower than 0. Thus, the *cdf* of the node lifetime is

$$F_{L(C)}(T) = \Pr\{L(C) \leq T\} = \Pr\{C - E(T) \leq 0\}. \tag{5.18}$$

As explained in Section 5.3.4, when T is large, $E(T) \sim \mathcal{N}(\mu(T), \sigma^2(T))$, where $\mu(T)$ and $\sigma^2(T)$ are given by Theorem 3. Thus, the *cdf* of single-node lifetime is approxi-

mated as

$$F_{L(C)}(t) \approx Q\left(\frac{\mu(t) - C}{\sqrt{\sigma^2(t)}}\right) . \tag{5.19}$$

5.4.2 Network Lifetime Distribution

Since every node needs to be alive during the network lifetime, the network lifetime (NL) distribution is obtained for a WSN with random deployment as:

$$F_{NL}(t) \approx 1 - \prod_{\boldsymbol{x}\in\mathcal{A}} (1 - p_{\text{ex}}(\boldsymbol{x}) \Pr\{L(\boldsymbol{x}, C(\boldsymbol{x})) \leq t\}), \tag{5.20}$$

where $L(\boldsymbol{x}, C(\boldsymbol{x}))$ is the lifetime for a node located at $\boldsymbol{x}$, if any, with battery capacity $C(\boldsymbol{x})$. Using the approximation in (5.19) for the single-node lifetime distribution, the network lifetime distribution is approximated by

$$F_{NL}(t) \approx 1 - \prod_{\boldsymbol{x}\in\mathcal{A}} \left(1 - p_{\text{ex}}(\boldsymbol{x}) Q\left(\frac{C(\boldsymbol{x}) - \mu(\boldsymbol{x}, t)}{\sqrt{\sigma^2(\boldsymbol{x}, t)}}\right)\right), \tag{5.21}$$

where $\mu(\boldsymbol{x}, t)$, $\sigma^2(\boldsymbol{x}, t)$ are given by Theorem 3 for the node located at $\boldsymbol{x}$. Moreover, $\mathcal{A}$ is the network area. To calculate the product, area $\mathcal{A}$ is discretized into small areas of size $\Delta\boldsymbol{x}$, and $p_{\text{ex}}(\boldsymbol{x})$ is the probability that there exist a node in the small area around $\boldsymbol{x}$, and is a function of the network density ρ. It is obtained by $p_{\text{ex}}(\boldsymbol{x}) = \rho\Delta\boldsymbol{x}$.

For a network with deterministic deployment containing N nodes, the network lifetime distribution, and its Normal approximation are obtained by

$$F_{NL}(t) = \Pr\{NL \leq t\} = 1 - \prod_{\boldsymbol{x}\in\mathbb{X}} \Pr\{L(\boldsymbol{x}, C(\boldsymbol{x})) \geq t\} \tag{5.22}$$

$$\approx 1 - \prod_{\boldsymbol{x}\in\mathbb{X}} Q\left(\frac{\mu(\boldsymbol{x}, t) - C(\boldsymbol{x})}{\sqrt{\sigma^2(\boldsymbol{x}, t)}}\right), \tag{5.23}$$

where $\mathbb{X}$ is the set of locations for all the nodes in the network.

A special case is also considered for the random deployment, where nodes are deployed in a circular plane of radius R, and generate a homogeneous amount of local traffic to a sink, which is located at the center of the plane. The battery capacity, C, for each node is the same. Moreover, each node forwards packets to neighbors closer to the sink. In this scenario, the energy consumption and lifetime analysis can take advantage of the symmetry of the topology as explained next.

The entire circular plane is discretized into concentric narrow rings with width Δr indexed by their distance to the sink, r. Each node senses the physical events, and generates packets with traffic rate λ_{lc}. By symmetry, the relay traffic $\lambda_{\text{re}}(r)$ is the same for all nodes in the same ring r. Hence the variables for a node in ring r are indexed by the distance r. Then, the distribution of the network lifetime, and its Normal distribution approximation are

$$\begin{aligned} f_{NL}(t) &= 1 - \prod_{r=0}^{R} \prod_{\theta=-\pi}^{\pi} \left(1 - p_{\text{ex}}(r) \Pr\{L(r,C) \leq t\}\right), \\ f_{NL}(t) &\approx 1 - \prod_{r=0}^{R} \prod_{\theta=-\pi}^{\pi} \left(1 - p_{\text{ex}}(r) Q\left(\frac{C - \mu(r,t)}{\sqrt{\sigma^2(r,t)}}\right)\right), \end{aligned} \tag{5.24}$$

respectively, where $p_{\text{ex}}(r) = \rho \Delta r \Delta \theta r$, and $\mu(r,t)$, $\sigma^2(r,t)$ are given by Theorem 3 for nodes in ring r.

5.5 Case Study: Anycast Protocol

In this section, the techniques in Sections 5.3 - 5.4 to capture the energy consumption and lifetime distributions are utilized for an anycast protocol, as described in Section

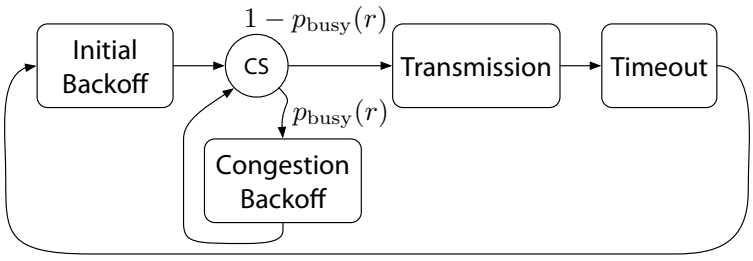


Figure 5.1: The process of transmitting beacon packets.

2.3.2. The anycast protocol studied in this section is the same as the one being studied in Section 3.6. Thus, the process of constructing the DTMC $\{X_n\}$ is based on the discussion in Section 3.6. The DTMC $\{X_n\}$ for the anycast protocol is presented in Figure 3.6. Then, the energy consumption distribution for each node is obtained accordingly. Finally, the lifetime distributions for each node and the network are found.

For the energy and lifetime analysis with anycast protocol, we assume that nodes are deployed in a circular plane of radius R, have a homogeneous battery capacity C, and generate a homogeneous amount of local traffic to a sink located at the center of the plane. Because of the symmetry, node-specific variables are the same for each narrow ring with radius r, and are indexed by r. In the following analysis, when there is no ambiguity, the subscript r in ring-specific variables is omitted.

5.5.1 Energy Consumption in Each State

At any time, a typical WSN node conducts one of the following communication tasks: transmission, listening, receiving, and sleeping. Listening and receiving is considered the same since most popular architectures, such as Mica2 [88] and TelosB [90], consume similar power for these tasks. We also ignore the energy consumed for the data packet transmission. This is a valid simplification because majority of the energy is

consumed for idle listening and beacon transmissions. It is validated by testbed and simulation results in Section 5.6 that for parameters of interests, these assumptions are accurate. Therefore, there are three types of states in $\{X_n\}$: Beacon transmission, Sleeping, and Listening. Nodes consume a specific amount of energy ε_v in each state v, as will be discussed in the following.

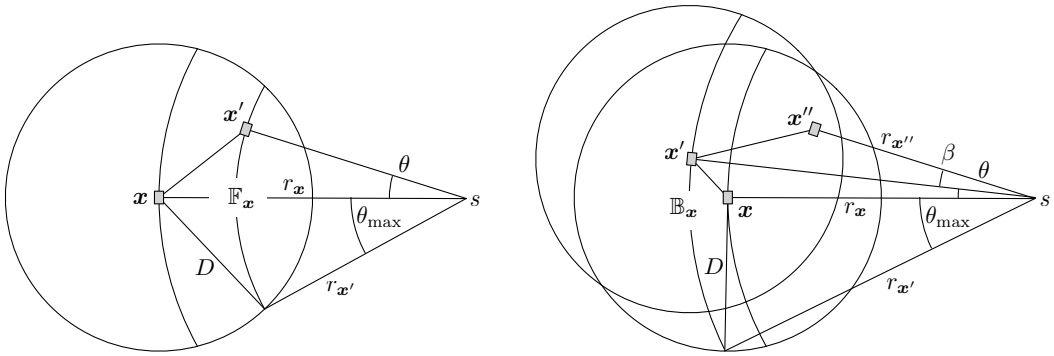
In practice, since battery voltage drops over time, battery capacity is often measured with normalized voltage. Therefore, energy is represented in the units of A·sec. In sleeping and listening states, the energy consumed during a time unit, $T_{\rm u}$, are $\varepsilon_{\rm sl} = I_{\rm sl}T_{\rm u}$, and $\varepsilon_{\rm li} = I_{\rm li}T_{\rm u}$, respectively, where $I_{\rm sl}$ and $I_{\rm li}$ are the measured current drawn from the battery in the sleep and listening modes, respectively.

The power consumption when the node is transmitting beacon packets, $\varepsilon_{\rm b}$, depends on the beacon transmission process shown in Figure 5.1. For every beacon packet, the node waits for a uniformly distributed random initial backoff with a maximum duration $T_{\rm ibo}^{\rm max}$, and whenever the channel is sensed busy before transmission, a congestion backoff is performed, which is also uniformly distributed with a maximum duration $T_{\rm cbo}^{\rm max}$. Then, the transmission takes a duration of $T_{\rm tx}$, which is determined by the packet size and the data rate. Finally, after the transmission, a timeout period of $T_{\rm to}$ is spent to wait for any possible CTS response. Therefore, the node transmits beacons only in a portion of time, and the portion, $\omega_{\rm b}$, should be obtained first to determine $\varepsilon_{\rm b}$. For a node within ring r, $\omega_{\rm b}$ is expressed as

$$\omega_{\rm b}(r) = \frac{T_{\rm tx}}{\left(\frac{T_{\rm ibo}^{\rm max}}{2} + \frac{T_{\rm cbo}^{\rm max} p_{\rm busy}(r)}{2(1-p_{\rm busy}(r))^2} + T_{\rm tx} + T_{\rm to}\right)}, \tag{5.25}$$

where $p_{\rm busy}(r)$ is the probability of sensing the channel busy, and is derived as follows.

First, as shown in Figure 5.2(a) (Figure 3.7 is redrawn here for convenience), the region within the transmission range of location $\boldsymbol{x}$, $\mathbb{C}(\boldsymbol{x})$, is divided into small



(a) Node $\boldsymbol{y}$ is in the feasible region of $\boldsymbol{x}$.

(b) Node $\boldsymbol{y}$ is in the infeasible region of $\boldsymbol{x}$, and node $\boldsymbol{z}$ is in the feasible region of $\boldsymbol{y}$.

Figure 5.2: The feasible region and infeasible region around node $\boldsymbol{x}$, divided into small areas. Figure 3.7 is redrawn here for convenience.

areas according to the polar coordinates centered at the sink. Each small area has a size approximated by $\Delta r \Delta\theta r$. Then, the probability of sensing the channel busy, $p_{\text{busy}}(r)$, is the probability that there is *at least* one node transmitting a packet in these areas. Considering that in WSN applications, sleeping cycles are usually long and duty cycles are usually very small, a sender node often has to wait for a relatively long period transmitting beacon packets before receiving a CTS response. Therefore, beacon packets are considered the dominant packets in the channel, and the major reason of a busy channel [49]. Thus, in the small area $(r : r + \Delta r, \theta : \theta + \Delta\theta)$, denote $p_{\text{ex}}(r)$ as the probability that there exists a node in this area, and $\phi_{\text{b}}(r)$ as the probability that at any time a node in this area, if it exists, is transmitting a beacon packet. Then $p_{\text{busy}}(r)$ is given by

$$p_{\text{busy}}(r) = 1 - \prod_{\boldsymbol{y}=(r',\theta')\in\mathbb{C}(\boldsymbol{x})} \Big(1 - p_{\text{ex}}(r')\phi_{\text{b}}(r')\Big), \tag{5.26}$$

where $p_{\mathrm{ex}}(r)$ is given by

$$p_{\mathrm{ex}}(r) = \rho \Delta r \Delta \theta r, \tag{5.27}$$

where ρ is the node density. The probability that a node in this area is transmitting a beacon packet, $\phi_{\mathrm{b}}(r)$, is given by $\phi_{\mathrm{b}}(r) = \pi_{\mathrm{b}}(r)\omega_{\mathrm{b}}(r)$, where $\pi_{\mathrm{b}}(r)$ is the total probability that the node is in one of the beacon transmission states in the DTMC $\{X_n\}$, and is given by adding the probabilities in the equilibrium state probability vector, $\boldsymbol{\pi}(r)$, corresponding to the beacon transmission states. Therefore, according to (5.26), for nodes located at $\boldsymbol{x}$ in ring r, the portion of time in which they transmit beacon messages, $\omega_{\mathrm{b}}(r)$, depends on its values for other nodes in its neighborhood, $\mathbb{C}(\boldsymbol{x})$. An iterative procedure is used for all r's to calculate $\omega_{\mathrm{b}}(r)$ at the end of Section 5.5.

Then, $\varepsilon_{\mathrm{b}}(r)$ is obtained by

$$\varepsilon_{\mathrm{b}}(r) = \Big(I_{\mathrm{li}}(1 - \omega_{\mathrm{b}}(r)) + I_{\mathrm{tx}}\omega_{\mathrm{b}}(r) \Big) T_{\mathrm{u}}, \tag{5.28}$$

where I_{li} and I_{tx} is the measured current when the node is listening and transmitting, respectively.

Finally, for this case study, we assume that the data processing time is far shorter than a time unit T_{u}. Since data processing is conducted when packets are generated or received, a fixed amount of energy, ε_{p}, is added to the energy consumption in the first state of each $\{C_n\}$.

5.5.2 Communication and Data Processing Energy Consumption

The other parameters in $\{X_n\}$, i.e., transition probability matrices and traffic rates in $\{I_n\}$ and $\{C_n\}$, are obtained according to the discussion in Section 3.6. Then, the equilibrium state probability vector, $\boldsymbol{\pi}(r)$, for the DTMC $\{X_n\}$ is obtained for each node $\boldsymbol{x}$. It should be noted that while we solve $\omega_{\rm b}(r)$, it is assumed that $\omega_{\rm b}(r')$ for all nodes $\boldsymbol{y}$ in range are known. This dependency is solved in an iterative manner. First, initial guesses of $\omega_{\rm b}(r)$ for all rings are set to all 0's in our evaluation. Then, updated values of $\omega_{\rm b}(r)$ are calculated. The iteration terminates when the difference between two consecutive iterations is negligible for each ring. Then, the energy consumption during a beacon time unit, $\varepsilon_{\rm b}(r)$, is obtained according to (5.28). Finally, the communication and data processing energy consumption distribution for any single node is calculated according to (5.9).

5.5.3 Topology Randomness

The variation of energy consumption and lifetime distribution introduced by topology randomness is captured by a zero-mean r.v. $E_{\rm tc}(T)$ with variance given in (5.10). In the following, the scaling coefficient, c, is empirically obtained for the anycast protocol for a node located in the ring r.

To calculate the scaling coefficient c, simulations are conducted with the anycast protocol. The energy consumption for each node are measured and the variance is recorded. The results are then used to find the scaling coefficient c using least squares regression. For given values of node density ρ, locally generated traffic rate $\lambda_{\rm lc}$, and

the duty cycle ξ, the scaling coefficient c is obtained by

$$\begin{aligned}\hat{c} &= \underset{c}{\operatorname{argmin}} \sum_{T\in\mathbb{T}} \left(\sigma^2(T) - \hat{T}\sigma^2_{\mathrm{cp},0} - cT^2\right)^2 \\ &= \frac{\sum_{T\in\mathbb{T}} \sigma^2(T)T^2}{\sum_{T\in\mathbb{T}} T^4},\end{aligned} \tag{5.29}$$

where $\hat{T}\sigma^2_{\mathrm{cp},0}$ is the variance of the energy consumed during T by communication and data processing, and is given by (5.12). $\mathbb{T}$ is a set of time durations in which the energy consumption is measured. It is set to $\{5, 10, 15, \cdots, 50\}$ hours in our studies.

Our empirical studies in Section 5.6.3 show that for the anycast protocol, the empirical expression for the scaling coefficient c is obtained as

$$c(\lambda_{\mathrm{lc}}, \rho) = a(\lambda_{\mathrm{lc}})^2\rho^{-2}, \tag{5.30}$$

where a is a constant irrelevant to λ_{lc}, ρ, and ξ. It is obtained by simulations for a single set of parameters as follows. For a random network with density ρ^* and locally generated traffic rate λ^*_{lc}, 100 realizations of topologies are generated. Then, the energy consumption is recorded, and the variance of energy consumption, $(\sigma^*)^2(T)$, during T is calculated for each $T \in \mathbb{T}$. The corresponding scaling coefficient c^* is then calculated according to (5.29) for $\lambda_{\mathrm{lc}} = \lambda^*_{\mathrm{lc}}$ and $\rho = \rho^*$. Thus, the constant a is obtained by solving (5.30) using λ^*_{lc}, ρ^*, and c^*.

Therefore, the variance of the topology compensation component is given by

$$\sigma^2_{\mathrm{tc}}(T) = a(\lambda_{\mathrm{lc}})^2\rho^{-2}T^2. \tag{5.31}$$

5.5.4 Total Energy Consumption and Lifetime Distribution

Finally, additional energy consumed by sensing activities are considered. The distribution of total energy consumption of the node is then obtained by (5.5) and (5.3).

With the energy consumption distribution for nodes in each ring known, the lifetime distribution for nodes in each ring, $L(r, C)$, is directly obtained by (5.19). Then, the distribution of the network lifetime, and its Normal distribution approximation are obtained by (5.24).

5.5.5 Extension to Other Protocols

The techniques in this section for the anycast protocol can be used to obtain the energy consumption and the lifetime distribution for other protocols, for example, TDMA protocols and RI-MAC [86]. First, the Markov chain for $\{X_n\}$ should be constructed according to the specific protocol behavior. Then, the single-node energy consumption distribution can be obtained by (5.9). Finally, the single-node and network lifetime distributions are found using (5.19) and (5.20), (5.21), respectively. The detailed solutions for other protocols are out of the scope of this work.

5.6 Analytical Results and Empirical Validations

To validate the accuracy of the proposed energy consumption and lifetime analytical framework, testbed experiments with 28 Crossbow TelosB motes and computer simulations using TOSSIM [56] are conducted. The distribution of single-node energy consumption, single-node lifetime, and the network lifetime for the anycast MAC protocol are calculated using the developed model and the asymptotic approximation. The results are then compared with the testbed experiments and simulations.

5.6.1 Experiment and Simulation Setup

5.6.1.1 Testbed Experiment Setup

Testbed experiments are conducted to validate the developed model for deterministic deployment, a special case for random deployment. Experiments for random deployment require at least hundreds of realizations of the random topology before valid statistical information can be gathered. Therefore, it is infeasible to validate our model solely using testbed experiments for random deployment. Testbed experiments are used to validate the model in small-scale deployments and validate the accuracy of computer simulations, which are then utilized to validate the proposed model for random deployment in larger scale and longer duration.

In the testbed experiments, nodes are placed in a circular area of radius $R = 4$ m, and the transmission power is set to -20 dBm, corresponding to a communication range of approximately 2.5m. A 1Ω resistor is placed in the circuit loop for each node, and the current drawn from the batteries of each node is obtained by measuring the voltage drop over the resistor. The voltage drop is measured using NI-USB 6210 DAQ modules [89] at 10kHz, converted to the current, and logged for 24 hours, as described in Section 2.4.1.2.

The values of radio, timing, and protocol parameters are listed in Table 5.1, whereas the parameters for channel model [107] used in the analysis and simulations are listed in Table 5.2.

5.6.1.2 Simulation Setup and Improvements

The computer simulations are performed using TOSSIM on FireFly [43]. To speed up TOSSIM simulations and obtain the simulation results for lifetime-scale durations, several techniques are utilized, as described in Section 2.4.2. First, the simulations

Table 5.1: List of radio, timing and protocol related constants and parameters.

Group	Notation	Description	Default Value
Radio	l_{p}	data packet size	40 bytes
	l_{m}	beacon and CTS message size	22 bytes
	R_{b}	channel bit rate	250 kbps
Timing	T_{p}	duty cycle period	10 s
	T_{a}	active period	5 s
	T_{b}	beacon transmission timeout	10 s
	$T_{\mathrm{ibo}}^{\mathrm{max}}$	maximum initial backoff	9.77 ms
	$T_{\mathrm{cbo}}^{\mathrm{max}}$	maximum congestion backoff	2.44 ms
	T_{tx}	data packet transmission time	1.6 ms
	T_{to}	beacon transmission interval	12 ms
Protocol	r_{th}	threshold radius	2.7 m
	ψ_{th}	threshold SNR	10 dBm

on 100 different topologies are conducted *in parallel* on different nodes of the supercomputer. Second, TOSSIM code is modified such that all *log and debug information is reduced*, except for the minimum necessary log on the energy consumption. This reduces the time spent on time-consuming I/O operations. Third, to further reduce the simulation time, the realistic channel model in TOSSIM is replaced by a *simplified channel model*, where packet transmissions are always successful if the sender-receiver distance is within a certain range, regardless of channel errors or collisions. The range is chosen such that at this distance, the average received signal SNR is equal to the SNR threshold used in the anycast protocol. This technique greatly reduces the time spent on communication simulations.

Each of the three techniques substantially increases the simulation speed for a

Table 5.2: List of channel-related constants and parameters.

Group	Notation	Description	Default Value
Channel	P_{n}	noise floor	-105 dBm
	$PL(D_0)$	pass loss at reference distance	52.1 dB
	D_0	reference distance	1 m
	η	pass loss exponent	3.3
	σ^{s}	standard deviation of log-normal fading/shadowing	5.5

Table 5.3: The simulation time and speed-up comparison for a 1-day simulation.

Number of Nodes		80	160	240
Network Density		0.028	0.057	0.085
Sim Time (s)	Original	4,848	10,205	17,633
	LogRedu	816	1,638	2,705
	LogRedu + SimpChan	56	83	103
Speed-up (%)	LogRedu	5.94	6.23	6.52
	LogRedu + SimpChan	86.6	123.0	171.2

typical network with radius $R = 30$ m and various number of nodes, as shown in Table 5.3. The speed-up is defined as the ratio between the simulation time required by the original simulation (Original) and when each of the speed-up technique is applied. The speed-up by parallelizing simulations on multiple computers is straightforward (equals to the number of computers used) and is not shown in the table. For the other speed-up techniques, log reduction (LogRedu) and simplified channel model (SimpChan), the time to simulate 1 day is shown in Table 5.3, as well as the speed-up ratio compared to the original simulation. The result reveals that both techniques

improve simulation speed greatly. By utilizing log reduction, simulation speed is increased by approximately 6 times. By utilizing log reduction and the simple channel together, the simulation speed is increased by around $87-171$ times. Moreover, with higher node density (160 nodes, and 240 nodes), the speed-up is more significant.

The first two techniques, parallelization and log reduction, increase simulation speed and introduce no error on the result. The simplified channel model, however, while further speeds up the simulation by more than 15 times, may introduce errors. The main reason is because by ignoring collisions, the simplistic channel model over-estimates the channel quality, and thus will introduce error in heavy traffic scenarios. This will be illustrated in Section 5.6.5.

In the following, we show that the developed framework is highly accurate using testbed experiments and simulations.

5.6.2 Validation of the Single-node Energy Analysis

We first analyze the energy consumption distribution in (5.9) for the deterministic deployment. The energy consumption distributions during $T = 60$ s for two nodes with distances of 2.6 m and 3.5 m to the sink are measured. The *cdf*s of the measured energy consumption are shown in Figure 5.3 with the analytical model results. It can be observed that the error of the analytical *cdf* is less than 5% compared to the empirical measurements for each node. It is also observable that the *cdf*s for the node at $r = 3.5$ m exhibit a steep increase at the energy level of 0.6 A·s. This is because there is a high probability that the node consumes exactly 0.6 A·s energy, which corresponds to the case where nodes are performing their normal duty cycle operations.

The same network topology is simulated using the original TOSSIM. The results

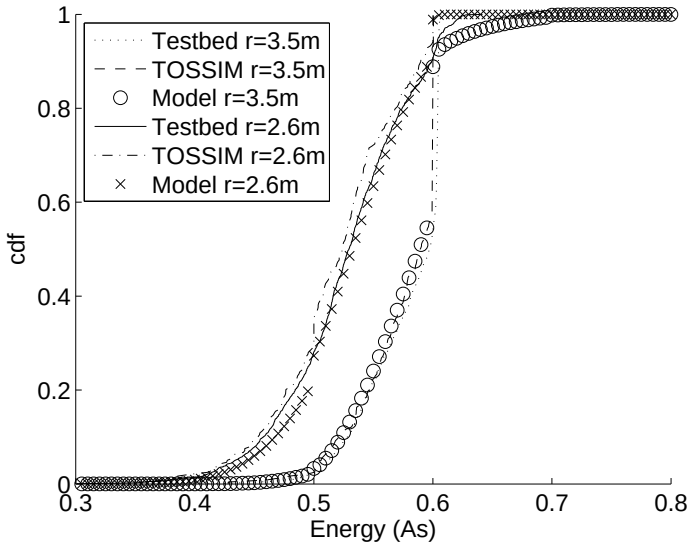


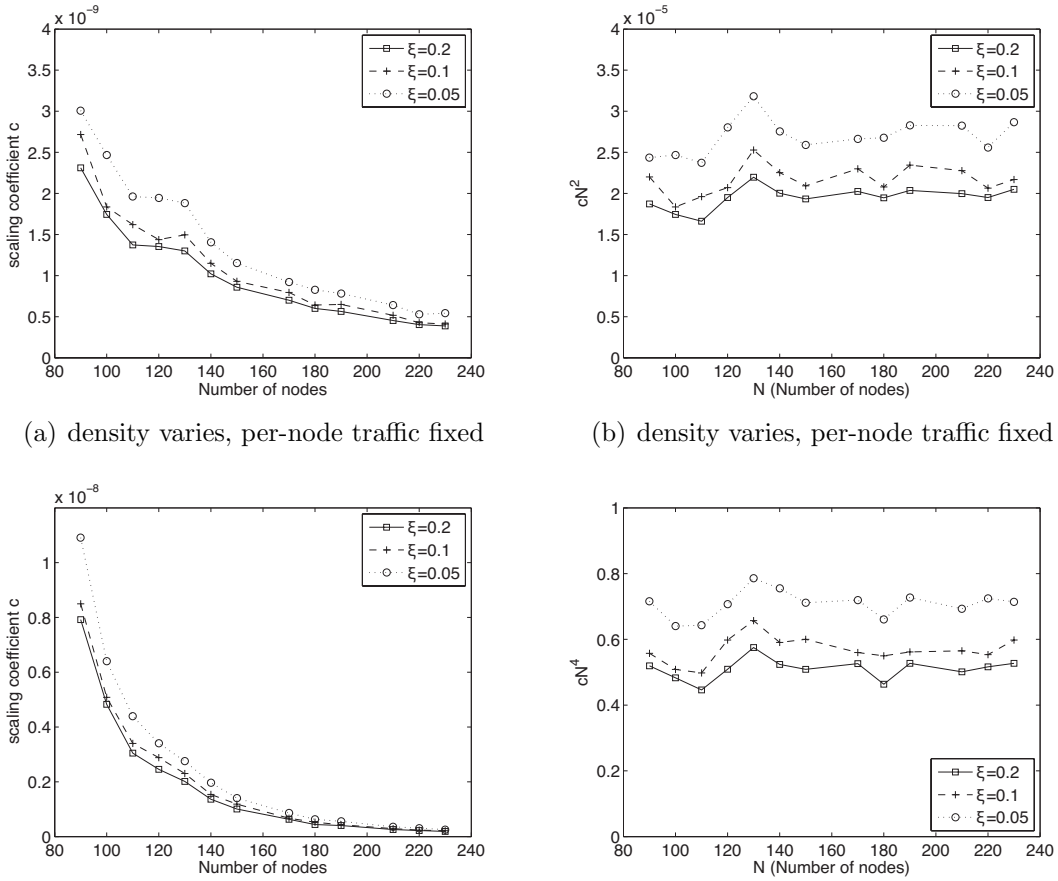
Figure 5.3: *cdf* of the energy consumption during 1 min. Testbed experiments, simulation, and analytical results are shown.

of the energy consumption distribution for each of the two nodes are also shown in Figure 5.3. The results suggest that the simulation results are also accurate compared to the empirical distribution with an error rate less than 5%. Therefore, in further experiments, we use simulation results to validate our model for random deployment networks in larger scale and longer duration.

5.6.3 Obtaining the Scaling Coefficient

In this subsection, the semi-empirical approach to obtain the topology compensation component for the anycast protocol is provided. Randomly deployed networks with PPP node locations are considered. Simulations are conducted to identify the relationship between the scaling coefficient c and network parameters, such as node density ρ, locally generated traffic rate λ_l, and the duty cycle ξ.

First, simulations are conducted with fixed duty cycle and traffic rate to evaluate the relation between c and the network density ρ. In the network with radius R, a



(a) density varies, per-node traffic fixed

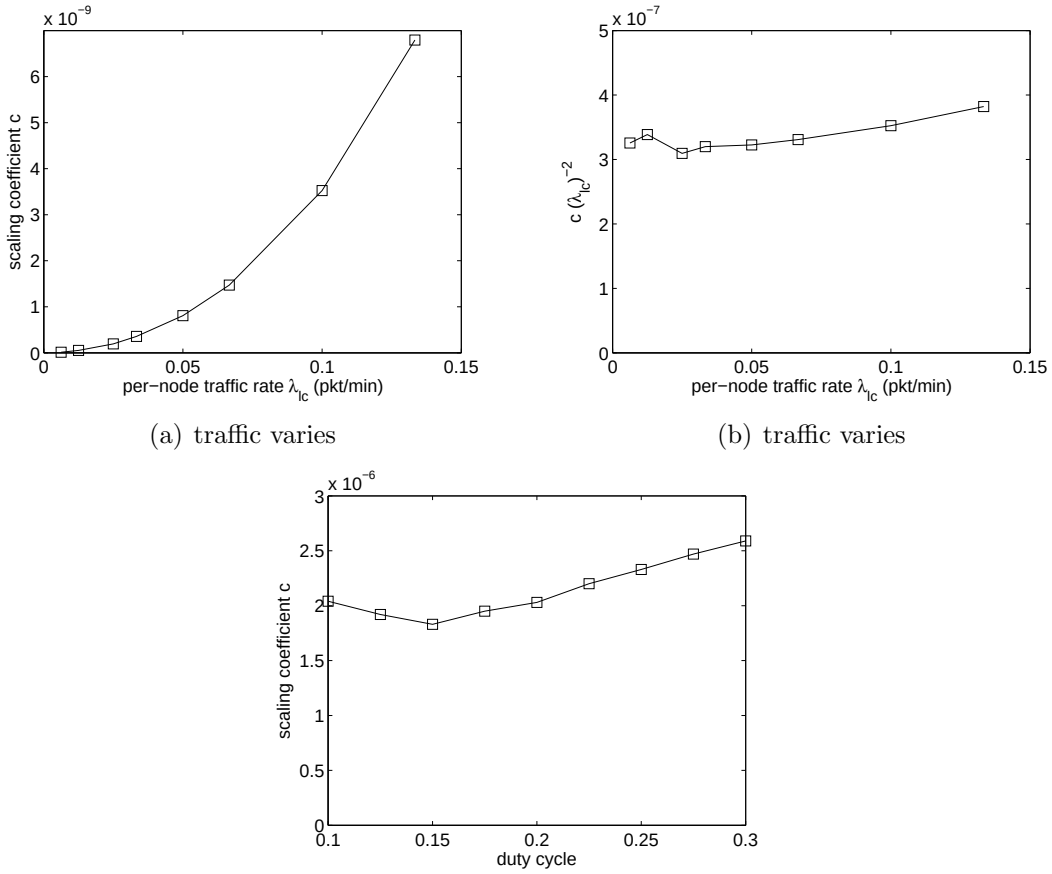
(b) density varies, per-node traffic fixed

(c) density varies, total traffic fixed

(d) density varies, total traffic fixed

Figure 5.4: The scaling coefficient c as a function of network density ρ in terms of total number of nodes N. In (a) the per-node traffic rate is fixed. In (c) the total traffic rate is fixed. cN^2 and cN^4 are used in (b) and (d) to reveal the empirical expression of c.

total of $N = 90, 100, \cdots, 230$ nodes are deployed uniformly (equivalent to Poisson point process). The duty cycle is $\xi = 0.2$, and the per-node locally generated traffic rate is $\lambda_{\text{lc}} = 0.05$ pkt/min. The value of c in each simulation is shown in Figure 5.4(a), and the value of cN^2 is shown in Figure 5.4(b). It is revealed that, the value of cN^2 is constant, and thus c is proportional to N^{-2}, or ρ^{-2}, when ξ and λ_{lc} are fixed. Note that the peaks in Figure 5.4(b) is due to the relatively small total number of random topology instances (i.e., 100). Higher number of instance would possibly smoothen



(a) traffic varies

(b) traffic varies

(c) traffic varies

Figure 5.5: (a) The scaling coefficient c and (b) the expression of $c\lambda_{\rm lc}^{-2}$ as functions of per-node traffic rate $\lambda_{\rm lc}$ to reveal the empirical expression of c. In (c) the scaling coefficient c is shown as a function of the duty cycle ξ.

the peaks, but leads to significantly higher computation cost in simulations.

Then, for each different total node number, the per-node traffic rate is varied, so that the total locally generated traffic rate is constant as 80 pkt/min, i.e., $\lambda_{\rm lc} = 80/N$ pkt/min. The value of c and cN^4 are shown in Figure 5.4(c) and 5.4(d), respectively. It can be observed that, the value of cN^4 is constant, suggesting that c is proportional to N^{-4}, or ρ^{-4}, when ξ is fixed, and $\lambda_{\rm lc} = 80/N$.

Simulations are also conducted for different values of per-node traffic rate $\lambda_{\rm lc}$, with fixed duty cycle $\xi = 0.2$, and fixed total node number $N = 160$. The value of c and

$c/(\lambda_{\text{lc}})^2$ are shown in Figure 5.5(a) and 5.5(b), respectively. It is shown that, $c/(\lambda_{\text{lc}})^2$ is constant, and thus, c is proportional to $(\lambda_{\text{lc}})^2$.

Finally, the effect of various values of duty cycle ξ is investigated, and is shown in Figure 5.5(c). It can be observed that, the variance of energy consumption does not change too much for different duty cycle values. Hence, in the empirical approach, the duty cycle is ignored when calculating the energy consumption variance.

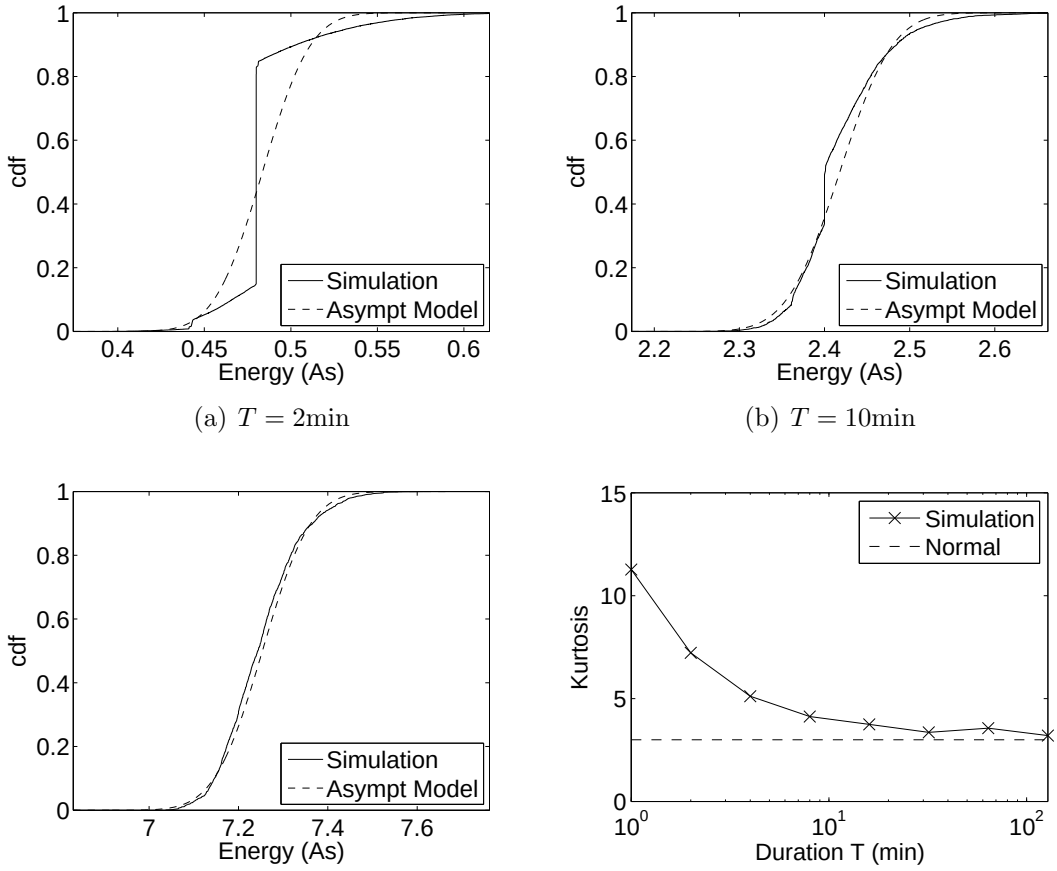
Then, all combined, the empirical expression for c is obtained as (5.30). Using the simulation result for $\rho^* = 0.057$ nodes/m^2, $\lambda^*_{\text{lc}} = 0.05$ pkt/min, and $\xi^* = 0.2$, the scaling coefficient c is expressed as

$$c = 3.1 \times 10^3 \rho^2 \lambda^2_{\text{lc}}. \tag{5.32}$$

5.6.4 Validation of the Normal Distribution Approximation

The asymptotic Normal distribution approximation of energy consumption for large T is validated using simulations. In the following simulations using the original TOSSIM, a network in a circular area with radius $R = 30$ m is assumed and the topology is randomly generated according to a Poisson distribution with density $\rho = 0.05$. Traffic rate is 0.1 pkt/min, $r_{\text{th}} = 10$ m, duty cycle is 0.2, and the transmission power is -15 dBm for all nodes. The threshold SNR is set to $\psi_{\text{th}} = 10$ dBm, and the communication range is 10m. In the following, we convert the network density ρ into the average node degree, i.e., the number of neighbor nodes in the communication range of each node, since it is a more intuitive presentation of density.

Each topology is simulated for 10 days, and 100 different topologies are generated. In addition, additional energy consumptions for sensing and data processing are added to simulate a fully operational WSN application. For each node, a sensor is powered



(a) $T = 2$min (b) $T = 10$min

(c) $T = 30$min (d) Kurtosis

Figure 5.6: *cdf* of the energy consumption during longer periods. As the duration increases, the energy consumption approaches the asymptotic Normal distribution.

for 5ms per 3 seconds, drawing a current of 10 mA. To process each generated and forwarded packet, each node draws an additional 5 mA current during 0.5 ms.

The *cdf* of energy consumption for node at $r = 27$ m for $T = 2$, 10, and 30 minutes are shown in Figure 5.6(a) - 5.6(c). The *cdf* of the asymptotic Normal distributions in Theorem 3 are also shown.

It can be observed in Figure 5.6(a) - 5.6(c) that as the duration increases, the energy consumption distribution converges to the asymptotic Normal distribution. We further validate the accuracy of the asymptotic approximation using Kurtosis [44,

94], which is a quantitative measure of the similarity between a particular distribution and the Normal distribution. For a r.v. z, Kurtosis is defined as

$$\kappa(z) = \frac{\mu_4(z)}{\sigma^4(z)}, \tag{5.33}$$

where $\mu_4(z)$ is the fourth moment of z, and $\sigma(z)$ is the standard deviation. The closer $\kappa(z)$ is to 3, the closer z is to a Normal distributed variable.

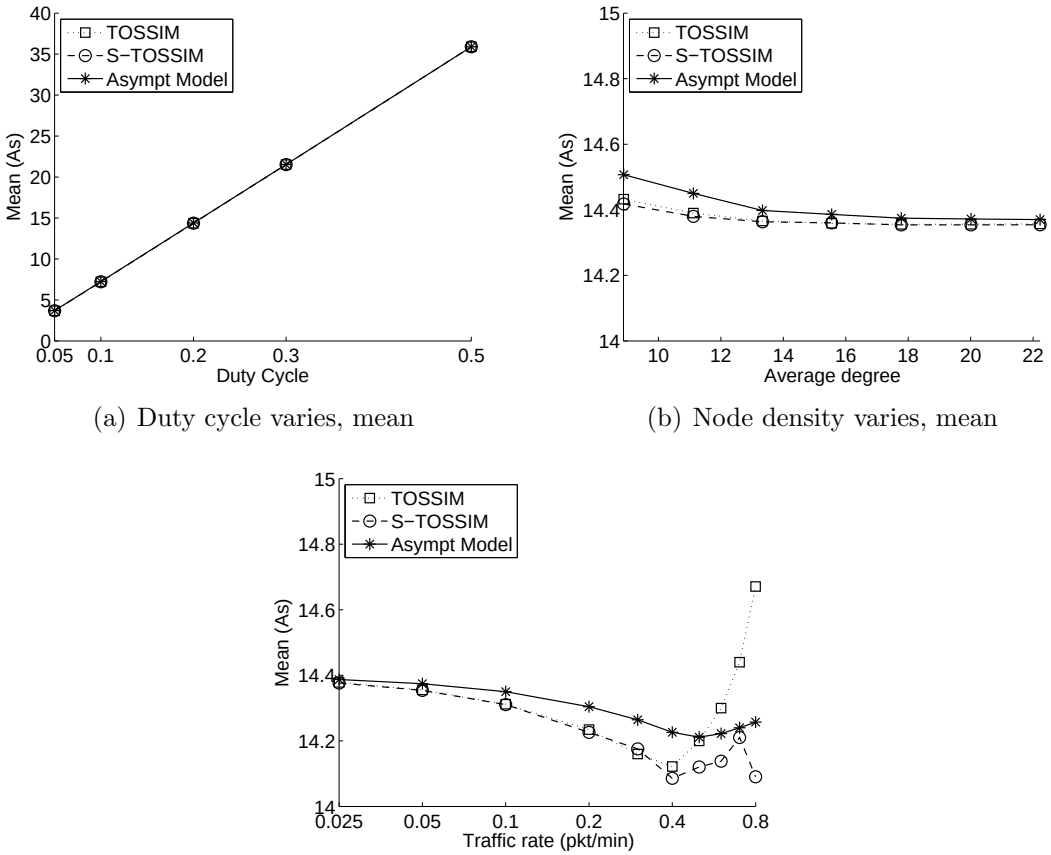
The energy consumption during 1, 2, 4, ..., 128 minutes for a node at $r = 27$ m are recorded to evaluate the Kurtosis. The results shown in Figure 5.6(d) reveals that for values of the duration T above 16 mins, the Kurtosis of the energy consumption converges to 3, which suggests that its distribution converges to the Normal distribution.

It is also found that generally the error of the mean and variance of energy consumption obtained by the model increases with r, the distance to the sink, compared to the experiment results. Thus, in the following experiments and simulations, we focus on a representative node at $r = 27$ m. This node has the highest error among nodes that are not affected by experiment and simulation artifacts introduced to nodes located at the rim of the network.

5.6.5 Model Validation with Different Network Parameters

To reveal how the accuracy is affected by different network parameters, simulations are conducted with various traffic rate, duty cycle, and node density, and the results are compared with the analytical results from the proposed framework. Both original TOSSIM channel model (TOSSIM) and simplified TOSSIM channel model (S-TOSSIM) are used. The default traffic rate is 0.05 pkt/min, the default duty cycle is 0.2, and the default node degree is 13.3. The mean and variance of the energy

consumption by a node 27 m apart from the sink during 1 hour are shown in Figures 5.7 and 5.8 for various parameters. Results provided by the analytical model are also shown.



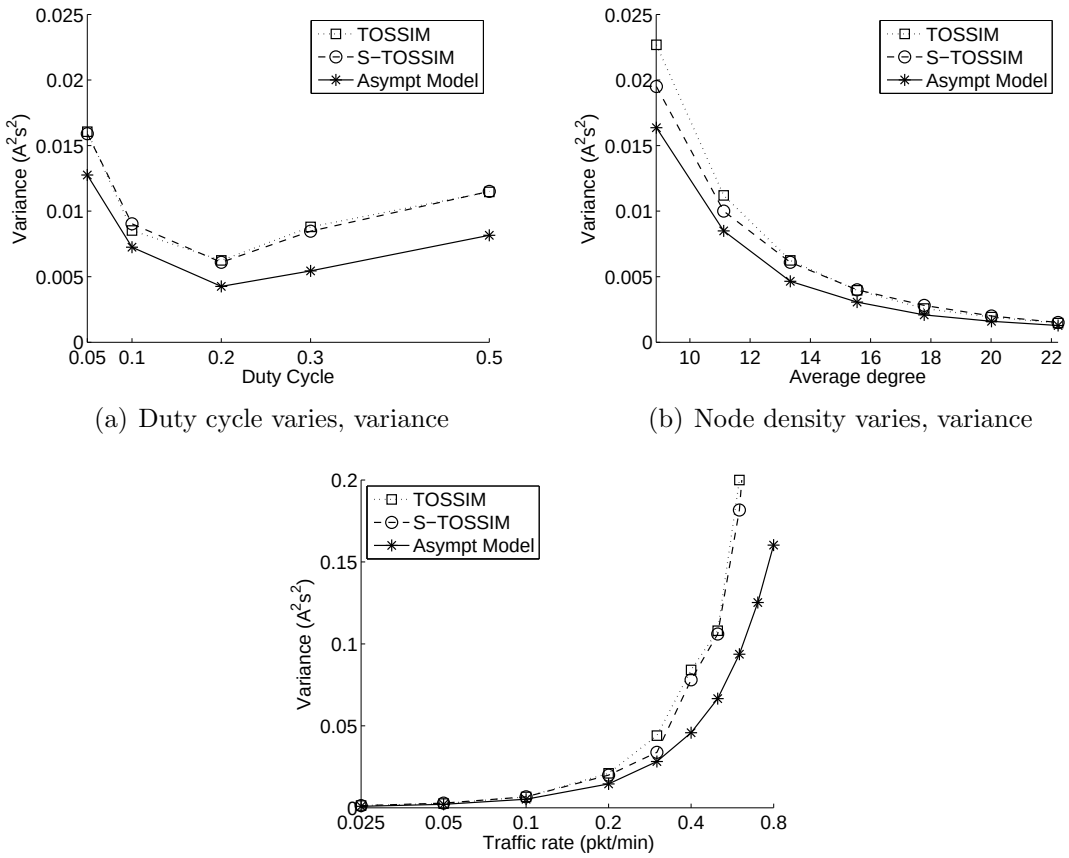
(a) Duty cycle varies, mean

(b) Node density varies, mean

(c) Traffic rate varies, mean

Figure 5.7: The mean of energy consumption during 1 hour for a node located at 27 m from the sink.

The mean energy consumption during 1 hour when only the duty cycle, the density, and the traffic rate is varied are shown in Figure 5.7(a) - 5.7(c), respectively. The corresponding variance of energy consumption are shown in Figure 5.8(a) - 5.8(c). It can be observed in Figure 5.7(a) that the energy consumption increases almost linearly with the duty cycle. Although the energy consumption is related to other network and protocol parameters as shown in Figure 5.7(b) and 5.7(c), the duty cycle



(a) Duty cycle varies, variance

(b) Node density varies, variance

(c) Traffic rate varies, variance

Figure 5.8: The variance of energy consumption during 1 hour for a node located at 27 m from the sink.

is the major factor that affects the energy consumption. In Figure 5.7(c), when the traffic rate is high, S-TOSSIM becomes less accurate due to the simplified channel model. Thus, the S-TOSSIM result is affected by simulation artifacts, which cause the S-TOSSIM curve to separate from the other results as the traffic rate increases.

On the other hand, the variance of the energy consumption is less sensitive to duty cycle. It fluctuates between around $0.005 - 0.015\text{A}^2\text{s}^2$ when the duty cycle changes from 0.05 to 0.5, as shown in Figure 5.8(a). However, the influence of network density and traffic rate is higher, as depicted in Figure 5.8(b) and 5.8(c). The variance is increased around 13 times when the node degree is reduced from 22.2 to 8.89, and

is increased around 180 times when the traffic rate is increased from 0.025 to 0.6 pkt/min, as explained in the following.

When the density increases, as depicted in Figure 5.7(b), generally the node consumes less energy on average, although this trend is less obvious when the node degree is higher than 13. This is because if the density is low, when transmitting beacon packets, each node needs to wait for a longer time before other nodes in the feasible region wake up, thus consuming more energy. The variance of energy consumption in Figure 5.8(b) is also decreasing, because higher density increases the chance of packets being transmitted with a short beacon transmission time. Thus, the variance of the beacon transmission time is low, leading to a low variance of energy consumption.

Finally, it can be observed in Figure 5.7(c) that when traffic rate increases, the mean energy consumption at first decreases for values lower than 0.5 pkt/min, and then increases for values higher than 0.5 pkt/min. The reason is that when each node is transmitting beacon packets, it does not respond to other beacon packets. Therefore, with a higher traffic rate, more nodes are transmitting, and fewer are available to send CTS responses. Thus, transmitting nodes need to wait for a longer time, and the energy spent on transmission is higher. With a moderate traffic rate, available relay nodes are enough, transmitting nodes can finish their transmissions and go to sleep early, effectively saving energy. With a lower traffic rate, however, the probability that nodes relay a packet and go to sleep early is low. Hence, energy consumption is higher than a moderate traffic rate.

On the other hand, as shown in Figure 5.8(c), the variance of energy consumption is monotonically increasing with higher traffic rate. This is because when the traffic is lighter, nodes are more likely performing quiescent operations and the beacon transmission periods are shorter. Thus, the activities are more homogeneous, and the variance of energy consumption is lower; when the traffic is heavier, nodes perform

beacon transmissions with a higher probability, and transmit for longer periods of time. Thus, nodes are more likely performing various activities, resulting a higher variance of energy consumption.

The results also suggest that the proposed framework provides accurate results for the mean of energy consumption with an error less than 3.5% (see Figure 5.7(a) - 5.7(c)). For the variance, the error is higher but still less than 21% (see Figure 5.8(a) - 5.8(c)). The only scenarios where the framework provides less accurate results are when the traffic rate is very high ($\geq$ 0.4 min/pkt). This is because when the traffic rate is high, the assumption that collision is negligible in Section 5.5.1 is no longer accurate. Nevertheless, since the majority of WSN applications operate with low traffic rate, the proposed framework is accurate in most of the scenarios.

Moreover, as expected, when the traffic rate is high, or when the network density is low, the simplified channel model in TOSSIM simulations becomes less accurate compared to the original channel model, as shown in Figures 5.7(b), 5.7(c), 5.8(b), and 5.8(c). This is because of the optimistic estimation of the channel condition when the density is low or the traffic is heavy. In the following, to speed up lifetime-scale simulations, only the simplified channel model is used and the network density and traffic rate are chosen to represent realistic WSNs scenarios, i.e., very heavy traffic or very low density scenarios are not analyzed.

5.6.6 Validation of Lifetime Distributions

In the following, the lifetime distribution analysis in Section 5.4 is validated using simulations. The network is assumed to be in a circular plane with a radius of 30 m, and contains 160 nodes. Each node generates data packets and transmits them to the sink located at the center of the network. The battery capacity is assumed to be

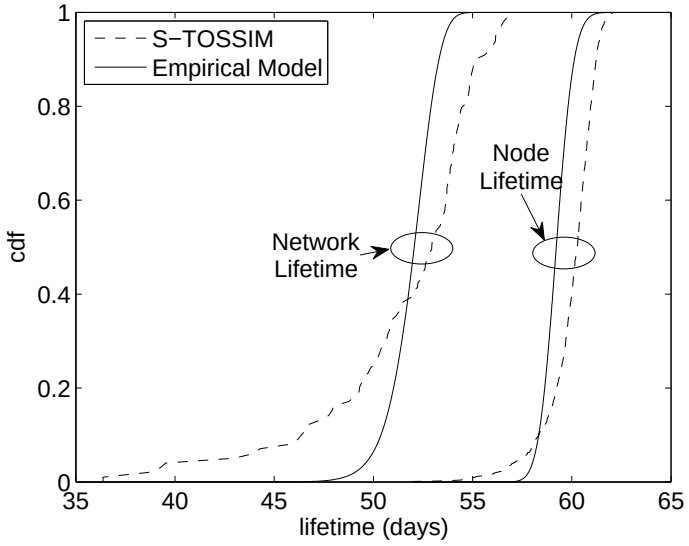


Figure 5.9: The network lifetime distribution and the lifetime distribution of a node at $r = 27$ m.

1500 mAh for all nodes. The locally generated traffic rate for all nodes is $\lambda_{\text{lc}} = 0.1$ pkt/min, and the duty cycle is $\xi = 0.05$. Other network and protocol parameters are the same as previous experiments.

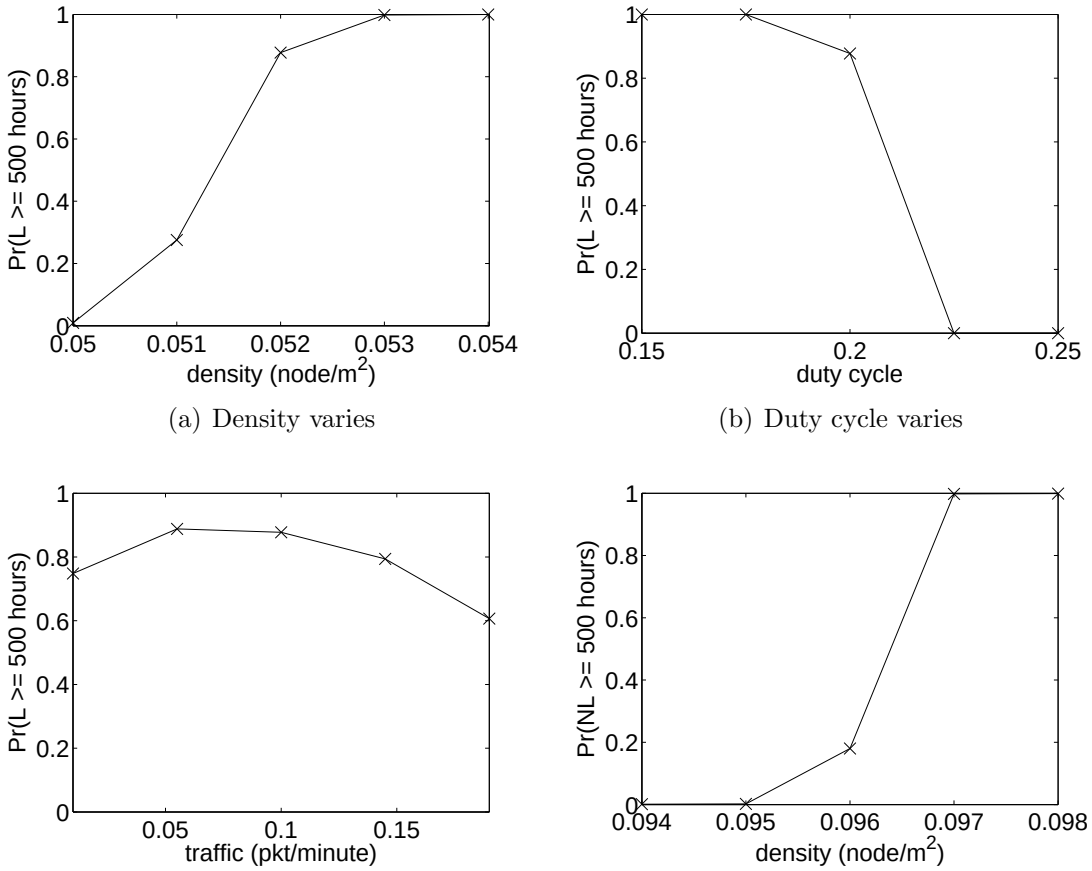
The single node lifetime distribution of a node located at distance 27 m to the sink, and the network lifetime distribution are shown in Figure 5.9. The result shows the probability to achieve a certain lifetime, or the lifetime achievable for a given probability. Compared to the simulations, for the same probability (higher than 0.2), the achievable lifetimes given by the proposed framework only have an error less than 3% for single node and 6% for the network. Considering that the desired probability in practice is usually higher than 0.5, the proposed framework yields very accurate results. The higher error of network lifetime is because the calculation of network lifetime requires the calculation of lifetime for all nodes in the network. Thus, any inaccuracy in node lifetime calculation will be accumulated and contributes to a larger error in network lifetime distribution.

On the other hand, the proposed framework outperforms simulations dramatically in terms of calculation speed. Even with multiple speed-up techniques described in Section 5.6.1, to determine the lifetime of a node in a typical setup, it requires about 3 hours for simulations with the simplistic channel model, and about 60 hours with the original TOSSIM channel model. In contrast, for any network setup in the experiments below, the analytical calculation requires only one computing unit, and takes less than a minute.

5.6.7 Network Design Observations

Next, as an example to show how the developed framework can be used to help network design, we investigate the relationship between the probability of achieving a given node or the network lifetime, and various network parameters, using the lifetime distribution obtained by (5.19). In each of the following tests, we consider a grid network. The network density ρ, the duty cycle ξ for all nodes, and the traffic rate λ_{lc} for all nodes are varied, respectively. The default values for these parameters are 0.052, 0.2, and 0.1 pkt/min. The network radius is 20 m. The battery capacities for all nodes are $C = 2000$ mA·H. Other parameters are kept unchanged from the previous experiments.

The probability that the lifetime of a node at distance $r = 12$ m is longer than 500 hours is shown in Figure 5.10(a) to 5.10(c). The results reveal that for the maximum probability of achieving this lifetime, the density should be no less than 0.053. It can be noticed that the probability increases dramatically from around 0 to almost 1 when the density changes within 0.002 node/m^2 from 0.095 node/m^2 to 0.097 node/m^2. This steep change is because the variation of network lifetime is small when the topology is deterministic. Moreover, reducing duty cycle directly reduces the energy



(a) Density varies

(b) Duty cycle varies

(c) Traffic rate varies

(d) Network lifetime, density varies

Figure 5.10: The probability of achieving a node lifetime (a, b, c) and a network lifetime (d) of 500 hours with various densities, traffic rates and duty cycles.

consumption, as observed in Figure 5.10(b). Finally, either increasing or reducing the traffic rate from 0.05 pkt/min results in a decrease of the probability of achieving this lifetime. The relationship between the probability and the network parameters are due to the relationship between energy consumption and the parameters, which are all explained in Section 5.6.5.

In the same network settings, the network lifetime distribution is also examined. The probability of achieving a 500 hours network lifetime (see (5.10)) is shown in Figure 5.10(d) for various network densities. To achieve this lifetime, the optimal

density is greater than 0.097, which is higher than the value for a single node lifetime guarantee in Figure 5.10(a). This is as expected because the network is only considered functional when *all* nodes are functional, which is a much stronger requirement than for a single functional node.

5.7 Conclusions

In this chapter, the probabilistic analysis of the energy consumption is provided. Energy consumption for communication, data processing, and sensing are all captured by the analytical framework. The energy consumption distribution for each node is derived. It is shown that, when the time duration is long, the energy consumption converges to a Normal distribution, and the mean and variance of such distribution are also provided. With the help of energy consumption distribution, the lifetime distributions for each node and the entire network are derived. The developed model is validated by both testbed experiments and TOSSIM simulations. The results show that the developed framework accurately models the distribution of the energy consumption and captures the randomness of multi-hop WSNs.

Chapter 6

Probabilistic Network Optimization

In this chapter, the results from the analysis models developed in Chapters 3, 4, and 5 are used to develop a probabilistic optimization framework for WSNs, which is used to demonstrate how to make decisions on choosing the optimal network parameters according to application requirements. The framework utilizes two types of probabilistic measures of QoS performance metrics: 1) for a given probability p, the performance metrics that can be achieved with at least probability p, and 2) the differences of performance metrics between two quantiles at p_1 and p_2. Given these probabilistic QoS performance constraints, the developed framework solves optimal network parameters for probabilistic QoS performance objectives. An anycast protocol is used to illustrate the application of this framework. Extensive evaluations are conducted to find the optimal network parameters for this protocol. Guidelines for designing networks and choosing optimal parameters for WSNs are provided using the optimization framework. It is also shown that mean analysis may lead to inaccurate results in network design due to lack of statistical information, which is intrinsically

provided by probabilistic analysis.

In this chapter, we first provide a study on related work in Section 6.1. In Section 6.2, the probabilistic optimization framework is formulated and a heuristic solution is presented. Then, in Section 6.3, a case study on the anycast protocol is provided, with a unified model for the end-to-end delay and the network lifetime based on the analysis in Chapters 3 and 5. Then, in Section 6.4, numerical results are presented from extensive evaluation experiments. Trends and insights that are not easily observable using traditional analyses are obtained. Finally, conclusions for this chapter are given in Section 6.5.

6.1 Related Work

Evaluating QoS provided by networks has been a major subject of research. QoS issues and techniques are intensively investigated for the the communication quality in traditional networks [7, 14, 19, 61, 62, 64]. For WSNs, the performance optimization has been a heated area of research ever since the concept of WSN was first introduced. The technique of Network Utility Maximization (NUM) has been applied to wireless networks and WSNs [16, 29, 75], and stochastic NUM is also proposed in [85, 102]. However, these studies mainly focus on the flow and throughput control of traffic in the network, and do not address optimization issues about other performance metrics. Other studies are focused on the probabilistic analysis of the delay in WSNs [8, 55, 57, 70, 101], and a few studies are conducted to investigate the probabilistic lifetime [71, 72]. While they provide statistical information for the performance metrics of concern, interrelationship among different performance metrics are left unexamined.

Multiple objective optimization problems are investigated in [10, 22, 50, 82, 84, 96, 103, 104]. Especially, the energy conservation and delay tradeoff problems are studied

in [10, 22, 50, 84, 104]. However, they do not provide a generic optimization framework to examine the relationship among other parameters, constraints, and objectives. Although a generic optimization framework is proposed in [31], like most of the other studies mentioned above, it does not fully capture the statistical characteristics about the QoS performance metrics in WSNs.

6.2 Probabilistic Optimization Framework

In this section, the probability optimization framework for WSNs is presented. We first discuss the probabilistic or deterministic objective and constraint functions, based on which the probabilistic optimization problems are formulated. For a set of given probabilistic or deterministic constraints on QoS performance metrics or network parameters, the goal is to find the optimal parameters, such that an objective QoS performance metric or a network parameter is minimized or maximized, depending on the application requirements. Finally, we present a multiple-local-search based technique to find the best estimation of the optimal solution.

6.2.1 Objective and Constraint Functions

Consider a particular probabilistic QoS metric $g(\boldsymbol{d})$, which is a random function of a set of design variables $\boldsymbol{d} = \{d_i : 1 \leq i \leq N_d\}$, where N_d is the number of design variables. We are interested in finding the following probabilistic characteristics:

Definition 7. *The* ***p-quantile*** *of a probabilistic QoS performance metric $g(\boldsymbol{d})$, denoted by $g^{(p)}(\boldsymbol{d})$, is defined as the value of $g(\boldsymbol{d})$ achieved with at least a probability of p.*

Definition 8. *The* $(p_{\mathrm{l}}, p_{\mathrm{h}})$***-quantile interval*** *of a probabilistic QoS performance metric* g*, denoted by* $g^{(p_{\mathrm{l}},p_{\mathrm{h}})}(\boldsymbol{d}), (p_{\mathrm{l}} \leq p_{\mathrm{h}})$*, is defined as the difference between* $g^{(p_{\mathrm{l}})}(\boldsymbol{d})$ *and* $g^{(p_{\mathrm{h}})}(\boldsymbol{d})$*, i.e.,* $g^{(p_{\mathrm{l}},p_{\mathrm{h}})}(\boldsymbol{d}) = g^{(p_{\mathrm{h}})}(\boldsymbol{d}) - g^{(p_{\mathrm{l}})}(\boldsymbol{d}), (p_{\mathrm{l}} \leq p_{\mathrm{h}})$.

The p-quantile is used to describe the value of the QoS performance with a probability guarantee, whereas the $(p_{\mathrm{l}}, p_{\mathrm{h}})$-quantile interval is used to describe how the value is "concentrated" or "spread", i.e., the predictability. For example, a lower $(0.1, 0.9)$-quantile interval of delay means that for the majority of the packets (all packets other than the fastest 10% and the slowest 10%), the delay is concentrated in a smaller region between the 10-quantile and the 90-quantile. Thus, the delay is easier to predict.

It is obvious that the p-quantile and $(p_{\mathrm{l}}, p_{\mathrm{h}})$-quantile interval are directly obtained from the *cdf*s of corresponding performance metrics. Given a probabilistic QoS metric $g(\boldsymbol{d})$, and its *cdf* $G_{g(\boldsymbol{d})}(g)$, the p-quantile and $(p_{\mathrm{l}}, p_{\mathrm{h}})$-quantile interval are given by

$$g^{(p)}(\boldsymbol{d}) = G^{-1}_{g(\boldsymbol{d})}(p), \tag{6.1}$$

$$g^{(p_{\mathrm{l}},p_{\mathrm{h}})}(\boldsymbol{d}) = G^{-1}_{g(\boldsymbol{d})}(p_{\mathrm{h}}) - G^{-1}_{g(\boldsymbol{d})}(p_{\mathrm{l}}), \tag{6.2}$$

respectively, where $G_{g^{-1}(\boldsymbol{d})}(g)$ is the inverse function of $G_{g(\boldsymbol{d})}(g)$. Obtaining the closed-from inverse function for $G_{g(\boldsymbol{d})}(g)$ in practice may be infeasible. In our evaluations, we first obtain the numerical expression of the *cdf* $G_{g(\boldsymbol{d})}(g)$, i.e., the probabilities of $g(\boldsymbol{d}) \leq g$ for a series of possible values of g are obtained as a series of tuples $(g_1, p_1), (g_2, p_2), \ldots$. Then, $G^{-1}_{g(\boldsymbol{d})}(p)$ is obtained by looking up the tuples for the ones with the closest probabilities, and its value is approximated using spline interpolation.

We also consider the deterministic measure of a set of QoS performance metrics, $f(\boldsymbol{d})$, for the scenarios where a probabilistic characterization of these metrics is unnecessary. The deterministic measure of these metrics are obtained as their expected

value.

6.2.2 Optimization Problem Formulation

More formally, consider a set of probabilistic QoS performance metrics $\mathcal{G} = \{g_j : 1 \leq j \leq N_{\mathcal{G}}\}$, and a set of deterministic QoS performance metrics $\mathcal{F} = \{f_k : 1 \leq k \leq N_{\mathcal{F}}\}$, where $N_{\mathcal{G}}$ and $N_{\mathcal{F}}$ are the number of probabilistic metrics and deterministic metrics, respectively. All of these metrics are functions of a set of design variables $\boldsymbol{d} = \{d_i : 1 \leq i \leq N_d\}$, where N_d is the number of design variables. Then, the constraints on a probabilistic performance metric g_j can be written as

$$g_{\mathrm{ql},j} \leq g_j^{(p_j)}(\boldsymbol{d}) \leq g_{\mathrm{qh},j}, \tag{6.3}$$

$$g_{\mathrm{vl},j} \leq g_j^{(p_{\mathrm{l},j},p_{\mathrm{h},j})}(\boldsymbol{d}) \leq g_{\mathrm{vh},j}, \tag{6.4}$$

where $g_j^{(p_j)}$ is the p_j-quantile of metric g_j, and $g_j^{(p_{\mathrm{l},j},p_{\mathrm{h},j})}$ is the $(p_{\mathrm{l},j}, p_{\mathrm{h},j})$-quantile interval of metric g_j. Moreover, $g_{\mathrm{ql},j}$ and $g_{\mathrm{qh},j}$ are the lower and upper *quantile requirements* on the p_j-quantile; $g_{\mathrm{vl},j}$ and $g_{\mathrm{vh},j}$ are the lower and upper *quantile interval requirements* on the $(p_{\mathrm{l},j}, p_{\mathrm{h},j})$-quantile interval.

The constraints on a deterministic performance metric f_k can be written as

$$f_{\mathrm{l},k} \leq f_k(\boldsymbol{d}) \leq f_{\mathrm{h},k}, \tag{6.5}$$

where $f_{\mathrm{l},k}$ and $f_{\mathrm{l},k}$ are the lower and upper deterministic requirements on metric f_k.

Accordingly, the optimization problem can be formulated as one of the following three.

6.2.2.1 Quantile Objective Optimization

In this type of optimization problem, the objective function is the p_o-quantile of some probabilistic metric $\mathcal{g}_o$, $(1 \leq o \leq N_{\mathcal{G}})$, i.e., the probability objective, where p_o is an application specified probability threshold:

$$\begin{aligned}
&\min_{\boldsymbol{d}} \ \mathcal{g}_o^{(p_o)}(\boldsymbol{d}) \ , \\
\text{or} \quad &\max_{\boldsymbol{d}} \ \mathcal{g}_o^{(p_o)}(\boldsymbol{d}) \ , && (6.6) \\
\text{given:} \ & d_{\mathrm{l},i} \leq d_i \leq d_{\mathrm{h},i}, && (1 \leq i \leq N_d), && (6.7) \\
& \mathcal{g}_{\mathrm{ql},j} \leq \mathcal{g}_j^{(p_j)}(\boldsymbol{d}) \leq \mathcal{g}_{\mathrm{qh},j}, && (1 \leq j \leq N_{\mathcal{G}}), && (6.8) \\
& \mathcal{g}_{\mathrm{vl},j} \leq \mathcal{g}_j^{(p_{\mathrm{l},j}, p_{\mathrm{h},j})}(\boldsymbol{d}) \leq \mathcal{g}_{\mathrm{vh},j}, && (1 \leq j \leq N_{\mathcal{G}}), && (6.9) \\
& \mathcal{f}_{\mathrm{l},k} \leq \mathcal{f}_k(\boldsymbol{d}) \leq \mathcal{f}_{\mathrm{h},k}, && (1 \leq k \leq N_{\mathcal{F}}), && (6.10)
\end{aligned}$$

where $d_{\mathrm{l},i}$ and $d_{\mathrm{h},i}$ are the lower and upper bound of the design variable d_i, respectively. Whether the problem is a minimization problem or a maximization problem depends on whether a smaller value or a larger value is desirable by the application. Generally, when $\mathcal{g}_o$ is the end-to-end delay, the energy consumption, or event detection delay, the problem is a minimization problem; when $\mathcal{g}_o$ is the network lifetime, the problem is a maximization problem.

6.2.2.2 Quantile Interval Objective Optimization

In the second type of optimization problem, the objective function is the $(p_{\mathrm{l},o}, p_{\mathrm{h},o})$-quantile interval of some probabilistic metric $\mathcal{g}_o$, $(1 \leq o \leq N_{\mathcal{G}})$, i.e., the reliability

objective, where $p_{\mathrm{l},o}$ and $p_{\mathrm{h},o}$ are application specified probability thresholds:

$$\begin{aligned} &\min_{\boldsymbol{d}} \ g_o^{(p_{\mathrm{l},o},p_{\mathrm{h},o})}(\boldsymbol{d}) \ , \\ \text{or} \quad &\max_{\boldsymbol{d}} \ g_o^{(p_{\mathrm{l},o},p_{\mathrm{h},o})}(\boldsymbol{d}) \ , \end{aligned} \tag{6.11}$$

where the constraints are the same as (6.7) - (6.10).

6.2.2.3 Deterministic Objective Optimization

In the last type of optimization problem, the objective function is one of the deterministic metrics $f_o, (1 \leq o \leq N_{\mathcal{F}})$:

$$\begin{aligned} &\min_{\boldsymbol{d}} \ f_o(\boldsymbol{d}) \ , \\ \text{or} \quad &\max_{\boldsymbol{d}} \ f_o(\boldsymbol{d}) \ , \end{aligned} \tag{6.12}$$

where the constraints are the same as (6.7) - (6.10). It should be noted that each design variable can be considered as a deterministic QoS performance metric. For example, the network density, ρ, is a design variable, but can also be a metric to minimize. In fact, when the network size is fixed, the network density can be regarded as a quality of service in broader sense, which describes the cost efficiency the network can provide.

6.2.3 Solution to the Optimization Problems

The solution to the aforementioned optimization problems is non-trivial. The major challenge is due to the generality of the network topology and communication protocols assumed for the analysis framework. Without *a priori* knowledge of the topology and the protocols, the objective function and constraint functions cannot

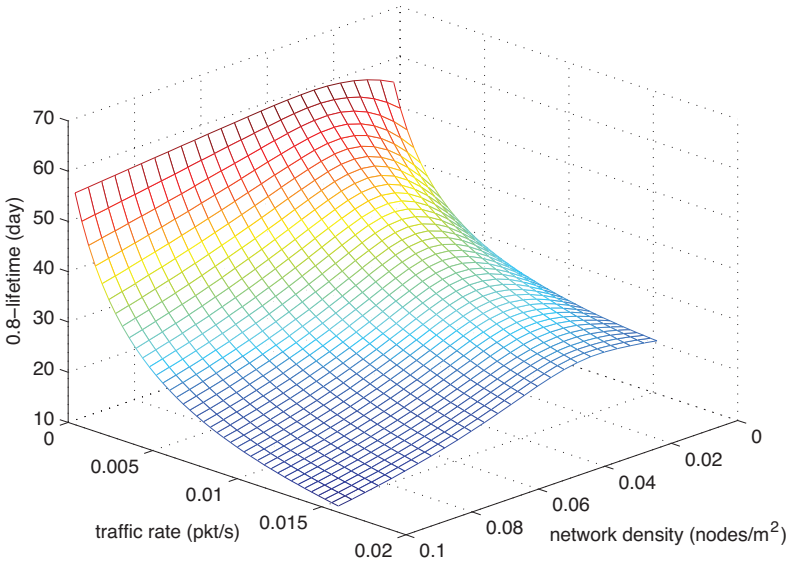


Figure 6.1: As a function of the network density and the traffic rate, the objective function, the 0.8-quantile of energy consumption is non-convex.

be considered convex, nor can they easily be converted to convex functions. In fact, our case study on a randomly deployed network with an anycast protocol, as will be discussed in Section 6.3, shows that the network lifetime is non-convex with respect to one of the design variables, the duty cycle, as depicted in Figure 6.1. Therefore, in the optimization framework, we use the following heuristic optimization technique.

Assume an optimization problem defined by (6.6), (6.11), or (6.12). The lower and upper bounds, $d_{\mathrm{l},i}, d_{\mathrm{h},i}$ on each design variable d_i form an orthogonal polyhedron, which is called the *parameter space* of the optimization problem. Each point within the parameter space corresponds to a vector of design variables that may or may not satisfy each of the constraint functions. The set of points in the parameter space that satisfies all the constraint functions is called the *feasible region*. The goal is to find the optimal objective function value within the feasible region.

In our proposed solution to the problem, N_{search} local-optimum searches are con-

ducted with a random initial search point. In each of the multiple searches, the initial search point is determined by sequentially choosing random points within the parameter space, until one point falls within the feasible region. Starting from this point, a derivative based local optimum search is conducted. Then, the global optimum is approximated by the best result in all the N_{search} optimum results found by each of the local searches. In the case when one or more of the local searches cannot converge due to non-convexity, these search procedures are terminated.

There are multiple benefits by utilizing this multiple-local-search technique. First, the technique is easy to implement, and does not require any form of prior knowledge about the topology and protocol. Second, the technique can be easily implemented taking advantage of multiple CPU cores or computers, since each of the local searches is totally independent of each other, thus can be run in parallel. Finally, when N_{search} is large, the optimum found by this technique asymptotically is always the global optimum, as the optimal solution will eventually coincide with one of the random initial points. It is easy to adjust the value of N_{search}, such that a trade-off can be made between the accuracy of result and the computation time efficiency.

6.3 Case Study: Randomly Deployed Network with Anycast Protocol

In this section, we discuss a case study for the purpose of illustrating the proposed analysis and optimization framework. For simplicity, we consider a special case with the anycast protocol and a randomly deployed network. The techniques used in this chapter can also be applied to networks with other protocols, and networks with deterministic deployment, which can be considered a special case of random

deployment with no topology variation.

6.3.1 Topology Model and the Anycast Protocol

We assume that nodes are deployed in a circular plane of radius R, and generate a homogeneous amount of local traffic to a sink, which is located at the center of the plane. The battery capacity, C, for each node is the same. Moreover, each node forwards packets to neighbors closer to the sink. Each node senses the physical events, and generates packets with traffic rate $\lambda_{\rm lc}$. By symmetry, the relay traffic $\lambda_{\rm re}$ is the same for all nodes with the same distance r to the sink. Hence the value of $\lambda_{\rm re}$, and other variables for these node are indexed by the distance r.

We consider a network utilizing a protocol with the anycast technique, as explained in Chapter 2. The analysis for the end-to-end delay and the network lifetime in this type of networks are described in Chapters 3 and 5, respectively.

6.3.2 Unified Probabilistic QoS Analytical Model

For the anycast protocol, a unified probabilistic analytical model is developed based on the frameworks developed in Chapters 3 and 5. This unified model captures the distribution of the end-to-end delay and the network lifetime in WSNs, as explained in the following.

We consider a 2-D network with a random deployment (the analysis also applies to the deterministic deployment) as described in Chapter 2. Each node is identified according to its location $\boldsymbol{x}$, and is characterized by its input traffic rate, $\lambda(\boldsymbol{x})$, queue length, $M(\boldsymbol{x})$, and battery capacity, $C(\boldsymbol{x})$. We also consider a log-normal fading channel model [107].

For the single-hop delay analysis and the single-node energy consumption analysis,

we combine the Discrete-Time Markov models in Chapters 3 and 5 by assuming that there is a single communication attempt for each packet. For protocols with multiple attempts, applying the proposed analysis techniques is trivial.

Based on this Discrete-Time Markov model, the end-to-end delay distribution and the network lifetime distribution can be found according to (3.37) in Chapter 3 and (5.21) in Chapter 5 for random deployment, or (3.35) in Chapter 3 and (5.22) in Chapter 5 for deterministic deployment.

6.3.3 Probabilistic QoS Optimization Problems

In the following, we define the set of design variables $\boldsymbol{d}$, the set of probabilistic metrics $\mathcal{G}$, and the set of deterministic metrics $\mathcal{F}$ for random deployed networks with the anycast protocol.

Although the design variables can be any variable that affects one or more performance metrics, in this work, for simplicity we only consider 3 design variables: the network density ρ, the locally generated traffic rate λ_{lc}, and the duty cycle ξ. We also consider the following probabilistic QoS performance metrics: the end-to-end delay from a node at the edge of the network to the sink, $t_{\text{e2e}}(R)$, and the network lifetime, NL. Note that in the circular plane network topology, the nodes at the edge will have the largest end-to-end delay. Thus, the end-to-end delay metric here is specifically the end-to-end delay from the node at the edge of the network area. Moreover, the deterministic performance metrics include TP, the traffic throughput received by the sink. In addition, we consider the network size to be fixed, thus the network density, ρ, is proportional to the total cost of the network, and is considered as a deterministic QoS metric.

Therefore,

$$\boldsymbol{d} = \{\rho, \lambda_{\text{lc}}, \xi\}, \tag{6.13}$$

$$\mathcal{G} = \{t_{\text{e2e}}(R), NL\}, \tag{6.14}$$

$$\mathcal{F} = \{TP, \rho\}, \tag{6.15}$$

where TP is calculated as

$$TP = \int_{0<r\leq R} 2\pi r \rho \lambda_{\text{lc}} p_{\text{deli}}(r) \mathrm{d}r, \tag{6.16}$$

where $p_{\text{deli}}(r)$ is the probability of final delivery of packets from nodes at r to the sink, and is given by $F_{e2e}(r, \infty)$, the *cdf* of the end-to-end delay from nodes at r to the sink, evaluated at $t = \infty$.

6.4 Numerical Results

In this section, the analysis and optimization framework developed in this chapter are evaluated for the randomly deployed network with the anycast protocol.

In our analysis, nodes are deployed in a circular plane with radius $R = 30$m, with a various network density from $\rho = 0.004$ to 0.1 nodes/m^2. Each node generates traffic at a rate from $\lambda_{\text{lc}} = 0.0004$ to 0.016 pkt/s. The duty cycle operation period is 10 s, with a duty cycle ranging from 0.25 to 1. The time unit is chosen as 0.25 s. The radio, timing and protocol related parameters are shown in Table 6.1, whereas the channel related parameters (refer to [107] for detailed explanations) are listed in Table 6.2. The transmission power is set to -10 dBm for all nodes. In the following, the numerical evaluations of the analysis framework is first presented, followed by the

Table 6.1: List of radio, timing and protocol related constants and parameters.

Group	Notation	Description	Default Value
Radio	l_{p}	data packet size	39 bytes
	l_{m}	beacon and CTS message size	22 bytes
	R_{b}	channel bit rate	250 kbps
Timing	T_{p}	duty cycle period	10 s
	T_{a}	wake period	2 s
	T_{b}	beacon transmission timeout	10 s
	$T_{\mathrm{ibo}}^{\mathrm{max}}$	maximum initial backoff	9.77 ms
	$T_{\mathrm{cbo}}^{\mathrm{max}}$	maximum congestion backoff	2.44 ms
	T_{tx}	data packet transmission time	1.6 ms
	T_{to}	beacon transmission interval	0.1 s
Protocol	r_{th}	threshold radius	10 m
	ψ_{th}	threshold SNR	10 dBm

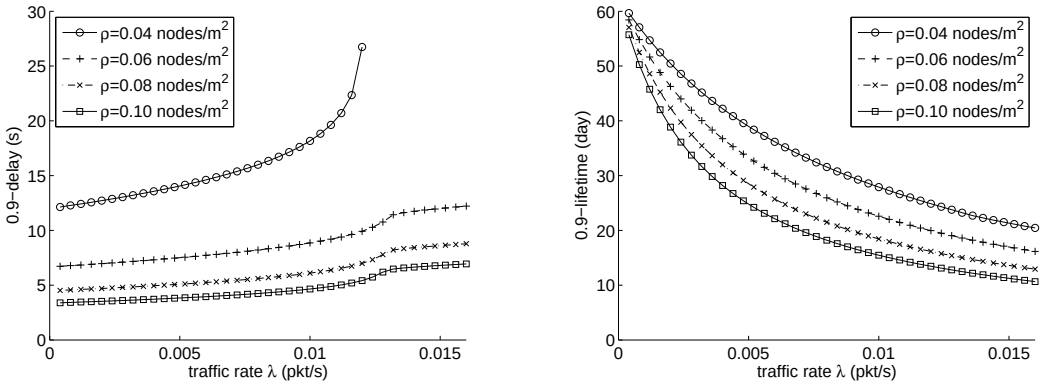
results from the optimization framework.

6.4.1 Numerical Analysis of Probabilistic QoS Metrics

In the first set of numerical evaluations, the analysis framework is used to evaluate the two probabilistic QoS metrics, the end-to-end delay and the network lifetime. Moreover, one deterministic metric, the throughput of traffic received by the sink, is also evaluated. As the foundations of the optimization framework, the results of the analysis are used to reveal the characteristics, including the trends and the convexity, of the probabilistic and deterministic measures of the performance metrics.

Table 6.2: List of channel-related constants and parameters.

Group	Notation	Description	Default Value
Channel	P_{n}	noise floor	-105 dBm
	$PL(D_0)$	pass loss at reference distance	52.1 dB
	D_0	reference distance	1 m
	η	pass loss exponent	3.3
	σ^{s}	standard deviation of log-normal fading/shadowing	5.5



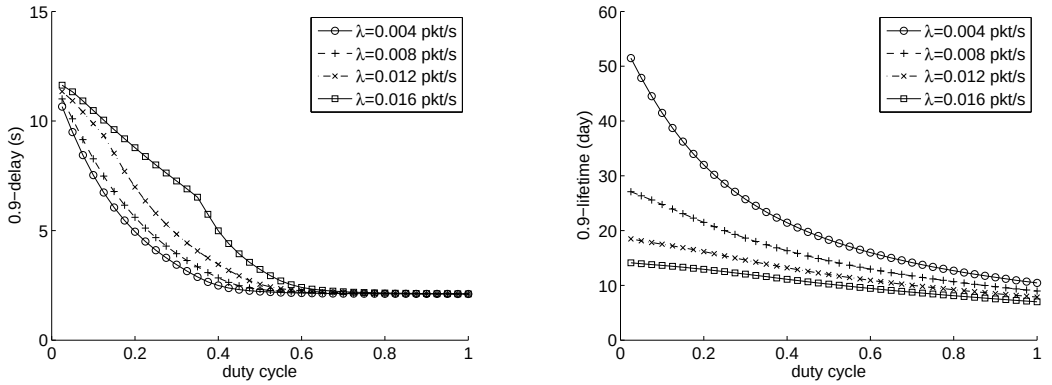
(a) 0.9-quantile of the end-to-end delay vs. traffic rate. Duty cycle is fixed as 0.2.

(b) 0.9-quantile of the network lifetime vs. traffic rate. Duty cycle is fixed as 0.2.

Figure 6.2: 0.9-quantile of the end-to-end delay and 0.9-quantile of the network lifetime as a function of traffic rate.

6.4.1.1 Probabilistic End-to-End Delay

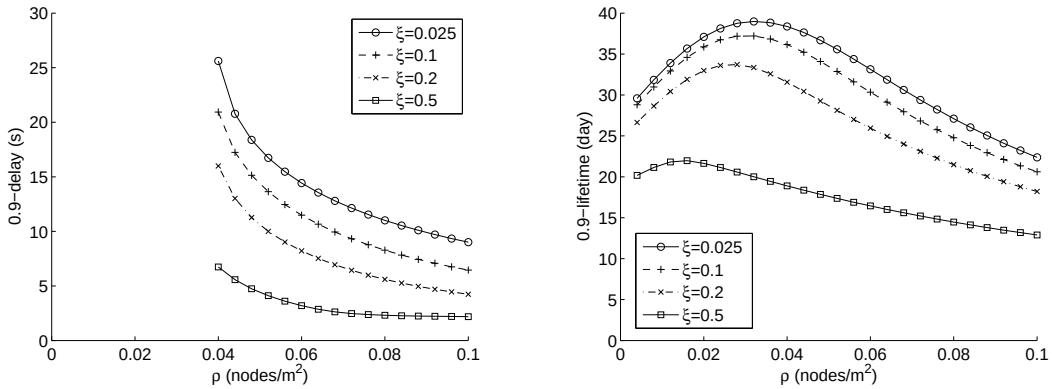
Figures 6.2(a), 6.3(a), and 6.4(a) depict the 0.9-quantile of the end-to-end delay from nodes at r to the sink (denoted as "0.9-delay" in the figures). Three design variables are examined: the locally generated traffic rate λ_{lc}, the duty cycle ξ, and the network density ρ. In each figure, one of these variables is varied, and the change of another variable is also shown using different curves. The last design variable for each figure remains fixed, as given in the captions.



(a) 0.9-quantile of the end-to-end delay vs. duty cycle. Network density is fixed as 0.08 nodes/m^2.

(b) 0.9-quantile of the network lifetime vs. duty cycle. Network density is fixed as 0.08 nodes/m^2.

Figure 6.3: 0.9-quantile of the end-to-end delay and 0.9-quantile of the network lifetime as a function of duty cycle.



(a) 0.9-quantile of the end-to-end delay vs. network density. Traffic rate is fixed as 0.008 pkt/s.

(b) 0.9-quantile of the network lifetime vs. network density. Traffic rate is fixed as 0.008 pkt/s.

Figure 6.4: 0.9-quantile of the end-to-end delay and 0.9-quantile of the network lifetime as a function of network density.

As can be observed from Figure 6.2(a), the 0.9-quantile of the end-to-end delay increases when the traffic rate increases, since higher traffic rate will cause higher queueing delay. Moreover, a lower network density will cause the delay to increase because less nodes will be in active states when each node starts to transmit. Thus, the waiting time is increased. Note that when traffic rate is high and network density

is low, the 0.9-quantile of delay does not exist. This is because less than 90% of packets are delivered to the sink from the edge nodes.

In Figure 6.3(a), it is shown that the 0.9-quantile of the end-to-end delay also decreases with a higher duty cycle. This is because higher duty cycle increases the number of active nodes when each node starts to transmit, thus reducing waiting time. It should be noted that the delay is less sensitive to the traffic rate when the duty cycle is either low or high. The reason is that when the duty cycle is low, the network is more likely to be in the saturated state where nodes keep transmitting without going into sleep or idle mode; when the duty cycle is high, nodes have a high chance to be immediately available for packet relaying. In both cases, increasing or decreasing the traffic rate does not change the delay too much.

Figure 6.4(a) shows that when the network density ρ is less than 0.04 nodes/m^2, the 0.9-quantile of the end-to-end delay does not exist. Therefore, when a highly reliable network in terms of end-to-end delay is desired, this analysis provides a guideline to determine network density.

Figures 6.2(a), and 6.3(a) also show that the 0.9-quantile of the end-to-end delay is generally not a convex function of the traffic rate or the duty cycle. For this type of non-convex performance measures, the developed heuristic solution is the most applicable approach we found.

6.4.1.2 Probabilistic Network Lifetime

The similar evaluations of probabilistic network lifetime are conducted and the results are shown in Figure 6.2(b), 6.3(b), and 6.4(b). It can be shown in Figure 6.2(b) that the 0.9-quantile of network lifetime decreases with higher traffic rate and higher network density. This is because higher traffic rate increases the energy nodes spend on beacon transmission, and higher network density increases the total traffic rate

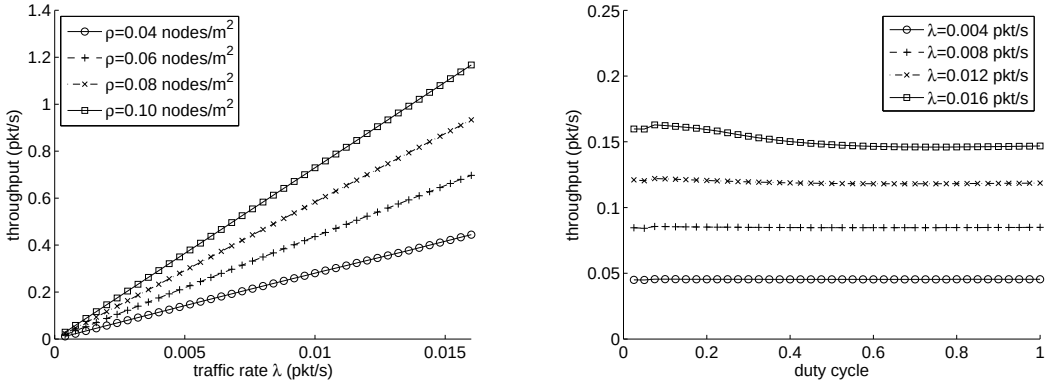
forwarded to the sink. Thus, the energy from nodes closer to the sink is drained faster. Note that when traffic rate is low, lifetime is not sensitive to the network density, as all nodes tend to perform a homogeneous default duty cycle operation.

Figure 6.3(b) shows that the 0.9-quantile of network lifetime decreases when the duty cycle increases, as expected. When the duty cycle is low and the traffic rate is low, nodes consume less energy and the network lifetime is high; when the duty cycle approaches to 1, the difference of lifetime diminishes among the networks with different traffic rate, because the energy consumption becomes dominated by idle listening in longer active periods. When the duty cycle is 1, all nodes remain active all the time, thus the difference of energy consumption solely comes from the power consumption used for different node activities. In our numerical evaluations, each node draws 0.2 mA current when it is listening, and 0.3 mA current when it is transmitting. Thus, the 0.9-quantile of the network lifetime is slightly higher for lower traffic rate, as nodes spend less energy on transmission, which is a higher energy consuming activity.

In Figure 6.4(b), it is shown that the 0.9-quantile of network lifetime has a peak when the density is around $0.15 - 0.3$ nodes/m^2, depending on the duty cycle. This is because when density is low, there is a higher chance that nodes are isolated from each other, and will spend more energy on continuously transmitting beacon messages. On the other hand, when the network density is higher, the total traffic forwarded to the sink is increased, thus the nodes close to the sink deplete their energy faster.

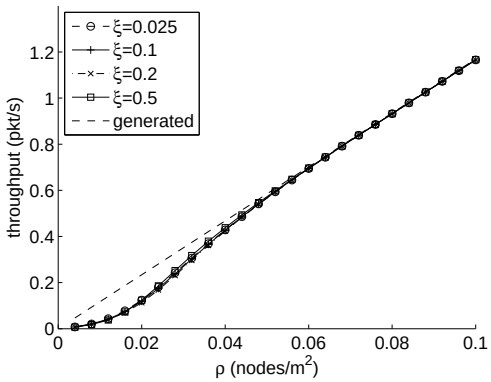
Figure 6.4(b) also shows that there is clearly a maximum point for each curve. The optimal network density ρ is different for different values of the duty cycle ξ. The developed heuristic solution can be used to accurately find the optimal density, as will be discussed in Section 6.4.2.

6.4.1.3 Throughput at the Sink



(a) Throughput vs. traffic rate. Duty cycle is fixed as 0.2.

(b) Throughput vs. duty cycle. Network density is fixed as 0.02 nodes/m^2.



(c) Throughput vs. network density. Traffic rate is fixed as 0.016 pkt/s.

Figure 6.5: The throughput as a function of traffic rate, duty cycle, and network density, respectively.

For the deterministic metric of throughput received at the sink, similar evaluations are also conducted, except that instead of the quantiles, the deterministic values of throughput are used. In Figure 6.5(a), it can be observed that the throughput almost increases linearly with generated traffic rate. However, Figure 6.5(c) shows that this is only the case for higher density, where almost every generated packet is delivered to the sink. When density is lower than 0.04 nodes/m^2, there is a larger portion of

packets that are not delivered to the sink, because when each node starts to transmit, it is less likely to find a neighbor that can relay packets. Moreover, these nodes will be more likely to build up a full queue, further preventing other nodes to use them as relay nodes. In both cases, there would be a higher chance of packet dropping.

It is interesting that Figures 6.4(a), 6.4(b), and 6.5(c) all show that network density lower than 0.4 nodes/m^2 would lead to negative impacts on the QoS performances. Thus, designers of the network should try to avoid such regions.

6.4.1.4 Probabilistic Measures

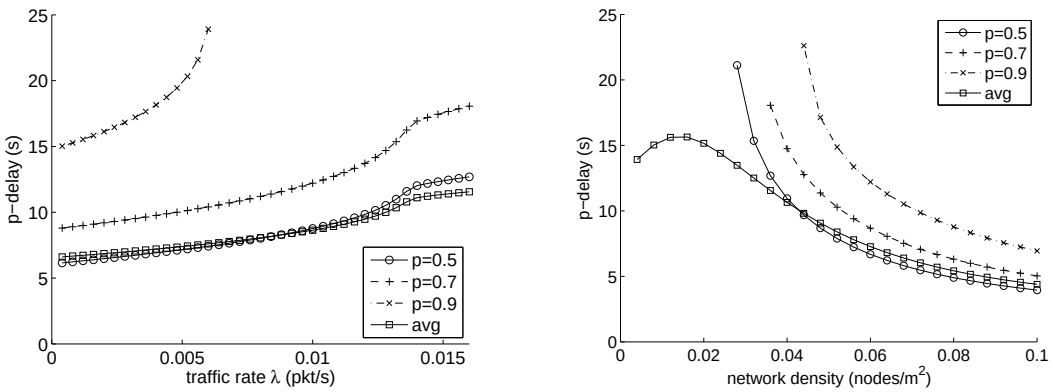
The analysis framework is also used to evaluate the probabilistic QoS performance metrics, the end-to-end delay and the network lifetime, in terms of their probabilistic measures.

Figure 6.6(a) shows the p-quantile of the end-to-end delay with p equal to 0.5, 0.7, 0.9, respectively, as a function of traffic rate. The three curves show the achievable end-to-end delay with these probabilities. In this evaluation, the network density is $\rho = 0.045$ node/m^2, and the duty cycle is $\xi = 0.2$. As a comparison, the average delay is also shown in the figure. The average delay is calculated as

$$\bar{t}_{\text{e2e}}(R) = \frac{\int_0^\infty t \cdot f_{\text{e2e}}(R,t)\mathrm{d}t}{\int_0^\infty f_{\text{e2e}}(R,t)\mathrm{d}t}, \tag{6.17}$$

where $f_{\text{e2e}}(R,t)$ is the *pdf* of the end-to-end delay from the edge of network to the sink. Note that the denominator in (6.17) is not necessarily always equal to 1, since in many cases a portion of packets will never reach the sink and their end-to-end delay is undefined, hence $\int_0^\infty f_{\text{e2e}}(R,t)\mathrm{d}t$ may be less than 1.

The change of average delay w.r.t. the traffic rate is similar to the trends of



(a) p-quantile of the end-to-end delay and the average delay vs. traffic rate.

(b) p-quantile of the end-to-end delay and the average delay vs. network density.

Figure 6.6: p-quantile of the end-to-end delay and the average delay vs. network density. Compared to the average delay analysis, the probabilistic delay analysis provides more accurate measurement of the end-to-end delay with statistical information.

0.5- and 0.7-quantiles of the delay. However, as the traffic rate increases, the average delay grows slower than the 0.5-quantile. This is because when traffic rate is higher, a larger portion of packets is lost. In this evaluation, when the traffic rate is higher than 0.006 pkt/s, more than 10% of packets are lost. The average delay is then calculated only for those packets that are eventually delivered. Therefore, the average delay does not contain the information of lost packets. When a high reliability of delay requirement is desired, the average delay may lead to opposite conclusion. This is further illustrated in Figure 6.6(b).

Figure 6.6(b) depicts the p-quantile of the end-to-end delay with p equal to 0.5, 0.7, 0.9, respectively, as a function of network density. The traffic rate is 0.016 pkt/s and the duty cycle remains to be 0.2. When the density is low, a large portion of packets is lost. In this evaluation, when the density is 0.044 nodes/m^2, at least 90% of packets are delivered from the edge of the network to the sink. When the density is lowered to 0.04 nodes/m^2, the percentage is decreased to values between 90% and 70%. When the density is further lowered to below 0.24 nodes/m^2, less than half of

the packets are delivered. It is expected that further reducing the network density would cause more packets to drop, and in many applications, this is a very inefficient use of communication resources, and should be avoided. However, according to the average delay shown in the figure, when density is low enough, the average delay would start to decrease, wrongfully suggesting a possible higher performance. In this case, the provision of statistical information makes probabilistic QoS analysis superior to average delay analysis.

6.4.2 Probabilistic QoS Optimization

The optimization framework is implemented using MatLab. As presented in Section 6.2, the optimization framework utilizes multiple local search procedures with random initial points to find possible local optima, and chooses the best solution among the local optima as the approximated global optimal solution. For the multiple local searches, each search procedure is implemented using the `fmincon` function provided by the optimization toolbox. The optimization algorithm used for `fmincon` is the interior point method. For each iteration in the local searches, the values of the optimization objective function and constraint functions are evaluated at points close to the current location to determine the estimated location for the next iteration. In the case where local searches cannot converge, a limit on the number of iterations, MAX_ITER, is enforced.

6.4.2.1 Brute Force Search Solution

Another optimization approach, the brute force search, is used for comparison. The objective function and constraint functions are evaluated at grid points in the entire design variable space. In our experiment, the ranges and increments of the three

design variables are selected as follows. The traffic rate, λ_{lc}, varies from 0.0004 pkt/s to 0.016 pkt/s with an increment of 0.0004 pkt/s; the duty cycle, ξ, varies from 0.025 to 1 with an increment of 0.025; the network density, ρ, varies from 0.004 nodes/m^2 to 0.1 nodes/m^2 with an increment of 0.004 nodes/m^2. Therefore, the design variable space is represented as a $40 \times 25 \times 40$ grid with $40,000$ points. The brute force search approach is to examine all these $40,000$ points, and find the point with maximum or minimum objective function while satisfying the constraints.

It should be noted that, since the evaluation of the objective function and constraint function values at each point would take approximately $15 - 30$ s, the total calculation time for all these $40,000$ points is approximately $7 - 14$ days. In comparison, with the search solution developed in Section 6.2, if 4 local searches are conducted sequentially, each with a maximum iteration of 25, the time needed is less than 2 hours. In our experiment, to expedite the brute force search, we utilize the supercomputer Firefly [43] located in Holland Computing Center at University of Nebraska-Lincoln to parallelize the calculation. In practice, especially when the dimension of the design variable space is higher, this brute force search approach is far from practical.

6.4.2.2 Accuracy of the Multiple Local Search

First, we evaluate the "accuracy" of the optimal solution found using the developed multiple local search technique. Specifically, several setups of the number of searches N_{search}, and maximum iteration allowed in each search, MAX_ITER, are examined.

For an optimization problem, we use the difference between the optimal solution found with each setup (of N_{search} and MAX_ITER) and the optimal solution found across all setups as the benchmark. It remains an open problem to find the *exact* global optimal solution to the probabilistic optimization problems in this dissertation.

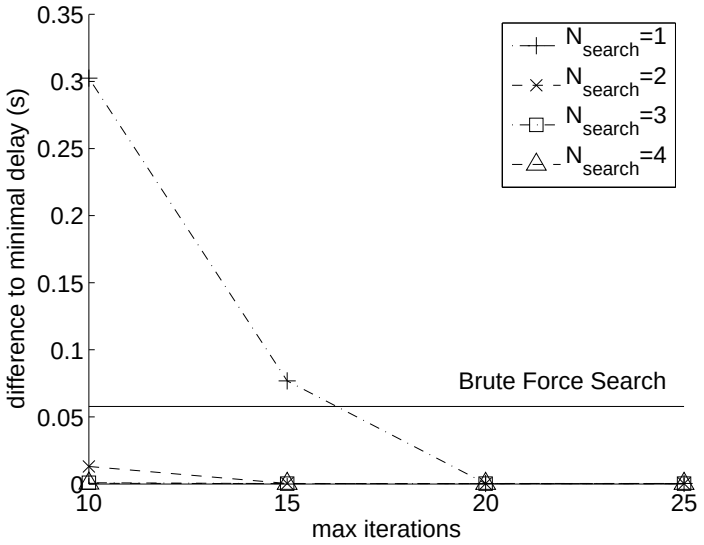


Figure 6.7: The difference between the optimal result found in each setup and the optimal result found in every setup. In each setup, the number of local searches, and the maximum iterations are varied. Optimal result from brute force search is also shown.

In Figure 6.7, the results of the difference for the following optimization problem are shown: given that the throughput received by the sink is higher than 0.5 pkt/s, and the 0.8-quantile of lifetime is longer than 30 days, the objective is to maximize the 0.9-quantile of the end-to-end delay. For each combination of N_{search} and MAX_ITER, 200 optimization procedures are conducted. Each procedure contains N_{search} local searches. A delay value, which is equal to or higher than the value found by 90% optimization procedures out of the 200, is shown in the figure.

The entire experiment contains 1600 local searches, out of which the best solution (difference to minimum delay is 0 in Figure 6.7, absolute value of the end-to-end delay: 2.29 s) is obtained as the benchmark. For comparison, the brute force search result for all $40,000$ points in the design variable space is also shown (difference to minimum delay is 0.0575 s, absolute value: 2.35 s). As can be observed, in all cases except when $N_{\text{search}} = 1$ and $MAX\_ITER \leq 15$, the developed solution yields a more

accurate result than the brute force search.

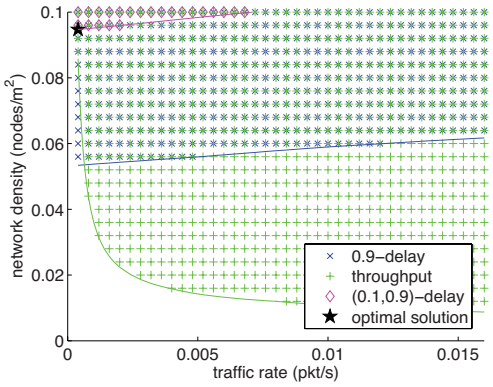
It can be observed that overall, the developed solution can find an accurate result. In the worst case, only one local search is conducted, and the search is terminated after 10 iterations. The solution found is as low as 0.3 s, or 13.1% higher than the benchmark solution. Moreover, as the number of searches N_{search} increases, or as the maximum iterations MAX_ITER for each search increases, the optimal solution found becomes closer to the benchmark solution. When $MAX\_ITER \geq 20$, or when $N_{\text{search}} \geq 3$, all optimization procedures can find the optimal result with negligible error.

6.4.2.3 Stochastic Optimization Aided Network Design

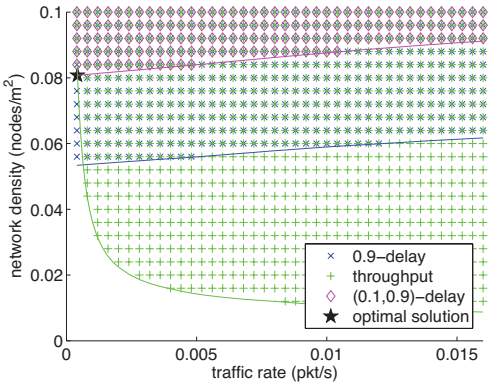
In the following, the results obtained by the developed optimization framework are used to aid the network design in two scenarios.

In the first scenario, the objective is to maximize the 0.8-quantile of the network lifetime while satisfying that: the throughput is higher than 0.1 pkt/s, the 0.9-quantile of the end-to-end delay is lower than 15 s, and the $(0.1, 0.9)$-quantile interval of the end-to-end delay is lower than a varying value from 6 s to 9 s. The optimal results found by the developed optimization framework are shown as a black star in Figure 6.8(a) - 6.8(c). It should be noted that only two dimensions (network density and traffic rate) of the design variable space are shown for clarity. The density-traffic rate plane is chosen in the 3-D space such that the corresponding duty cycle is the optimal duty cycle found by the optimization framework.

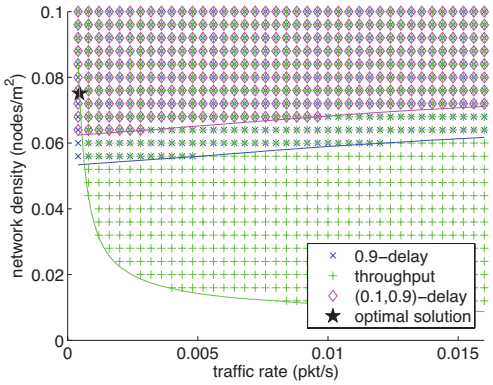
In the density-traffic rate plane, we also highlight the points satisfying one or more constraint functions by different markers. The values of constraint function values are obtained from the brute force search, thus the highlighted points form a grid in the density-traffic rate plane. Moreover, the boundaries of the region satisfying each of



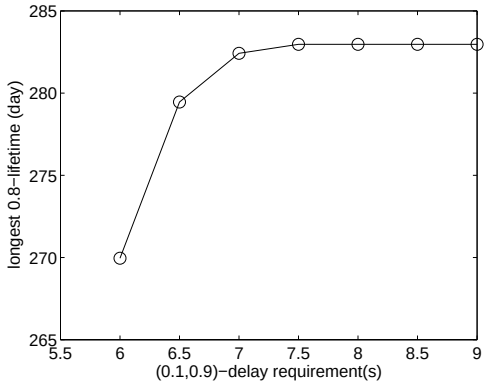
(a) $(0.1, 0.9)$-quantile interval= 6 s.



(b) $(0.1, 0.9)$-quantile interval= 7 s.



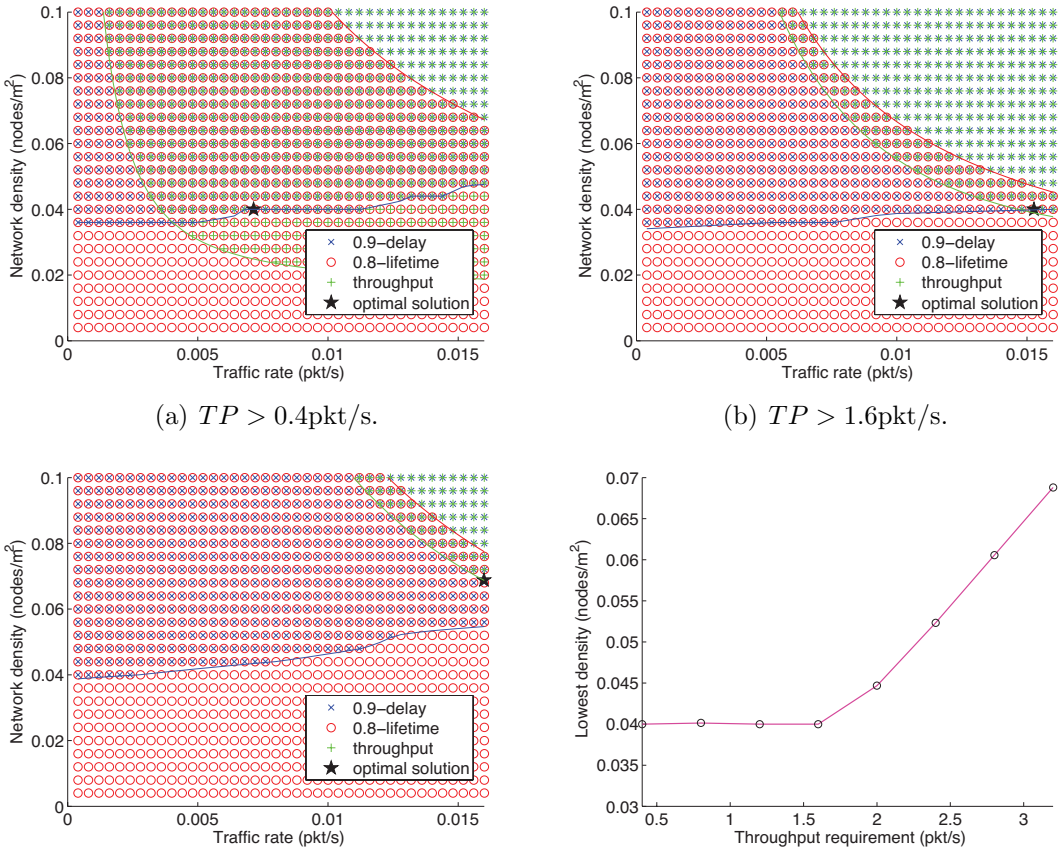
(c) $(0.1, 0.9)$-quantile interval= 9 s.



(d) longest 0.8-quantile vs. $(0.1, 0.9)$-quantile interval requirement.

Figure 6.8: Optimal network lifetime with varying $(0.1, 0.9)$-quantile interval requirement.

the constraint functions are illustrated using the MatLab function `contour`. Finally, the optimal 0.8-quantile of the network lifetime as a function of $(0.1, 0.9)$-quantile interval requirement is shown in Figure 6.8(d). The result suggests that a stricter $(0.1, 0.9)$-quantile interval requirement leads to a lower 0.8-quantile of the network lifetime. However, relaxing the $(0.1, 0.9)$-quantile interval requirement to higher than 7 s would not change 0.8-quantile of the network lifetime too much. This relationship is useful when a tradeoff between the network lifetime and the variation of the end-to-end delay must be made.



(a) $TP > 0.4$pkt/s.

(b) $TP > 1.6$pkt/s.

(c) $TP > 3.2$pkt/s.

(d) Minimum density vs. throughput requirement.

Figure 6.9: Optimal network density with varying throughput requirement.

In the second scenario, the objective is to find the lowest network density – and thus, lowest deployment cost – while satisfying: the 0.9-quantile of the end-to-end delay is lower than 15 s, the 0.8-quantile of the network lifetime is longer than 15 days, and the throughput is higher than a varying threshold varying from 0.4 to 3.2 pkt/s. The optimal results found by the framework are shown in Figures 6.9(a) to 6.9(c), and the optimal density as a function of throughput requirement is shown in Figure 6.9(d). The result shows that a lower (relaxed) throughput requirement would cause the lowest density and the total cost of the network to decrease. However, relaxing the throughput requirement further below 1.6 pkt/s would have almost no impact on the

lowest density. This is because when the throughput requirement is higher than 1.6 pkt/s, the optimal density is dominantly determined by the throughput requirement (Figure 6.9(c)), but when the throughput requirement is lower, the optimal density is dominantly determined by the end-to-end delay requirement (Figures 6.9(a) and 6.9(b)). This is extremely helpful in network design when a tradeoff must be made between the throughput and deployment cost.

6.5 Conclusions

In this chapter, an optimization framework for probabilistic QoS performance metrics is developed. Rather than traditional QoS analysis methods such as mean analysis, the optimization framework utilizes quantile-based QoS measures obtained from the analysis models in Chapters 3, 4, and 5 of this dissertation. In the framework, the probabilistic optimization problems are formulated and a heuristic solution is presented. A case study on the anycast protocol is then provided to show the application of this framework. Extensive numerical results reveal the relationship between QoS performance and network parameters. It is also shown that mean analysis may lead to inaccurate results in network design due to the lack of statistical information, which is intrinsically provided by probabilistic analysis. Numerical results also show that the developed optimization framework can be used to choose optimal parameters for networks.

Chapter 7

Dissertation Conclusions

In this chapter, we conclude this dissertation by summarizing the contributions and highlighting a few directions for future research.

7.1 Dissertation Contributions

To the best of our knowledge, this dissertation is the first work that systematically investigates the probabilistic QoS performance metrics in WSNs. The contributions of this dissertation are listed in the following.

7.1.1 Formal Definitions of Probabilistic QoS Performance Metrics

One of the aims in this dissertation is to formulate formal definitions of probabilistic QoS performance metrics in WSNs. The metrics discussed in this dissertation are: the end-to-end delay distribution, the network lifetime distribution, and the event detection delay distribution. From the distributions of these metrics, two probability

measures for each of these metrics, i.e., the p-quantile measure and the $(p_{\mathrm{l}}, p_{\mathrm{h}})$-bound, are defined.

7.1.2 Analytical Framework to Evaluate the Probabilistic QoS Metrics

A probabilistic analytical framework is proposed to evaluate the QoS performance metrics in two levels. At the node level, a Discrete-Time Markov queueing model is utilized to investigate probabilistic QoS performance metrics for individual nodes or hops. The single-hop delay distribution, the single-node energy consumption distribution, and the single-node lifetime distribution are derived at this level.

At the network level, the major challenge is the complexity and non-tractability of random factors in practical WSNs. Thus, based on the node level analysis, fluid models are further utilized to simplify the random factors and analyze probabilistic QoS performance metrics, including the end-to-end delay distribution, the network lifetime distribution, and the event detection delay distribution. Extensive testbed experiments and computer simulations are conducted to validate the accuracy of the framework.

7.1.3 Investigation on Relationship between Network Parameters and the QoS Performance Metrics

Using the proposed analytical framework, an optimization framework is also proposed to derive the optimal network and protocol parameters, subject to given probabilistic QoS performance requirements. This optimization framework is used to investigate the optimal network parameters, such as node density, traffic generation rate, and node duty cycle, such that one of the performance metrics or network parameters

is minimized or maximized, while a set of probabilistic QoS metrics constraints are satisfied. This framework is then used to aid the design and evaluation of network parameters and protocols before actually deploying the networks.

7.2 Future Research Directions

The results obtained in this dissertation suggest several potential research directions in future, as listed below.

- **Capturing More Dynamic Elements in WSNs.** With technology advances in both hardware and software, WSNs will most likely be equipped with more dynamic mechanisms in the future than those being investigated in this dissertation. For example, the hardware of sensor nodes may include more dynamic features such as wake-on-radio and environment energy harvesting, whereas the software may include routing protocols that are adaptive to the environment, and in-network processing such as data aggregation and dissemination. Other dynamic elements include: traffic patterns such as time-of-day dependent traffic, networks with multiple sinks, networks with mobile nodes, and collaborated sensing and actuation. Capturing these dynamic elements would make the probabilistic analytical framework more useful in practice.

- **Capturing More Complex Events.** For the event detection delay analysis in this dissertation, events are considered to occur at single points and are isolated from each other. In practice, the occurrences of events are often more complex. As future work, the event detection delay for simultaneous multiple events are an important research direction. These simultaneous events can occur in isolated spatial locations, or their detection ranges may overlap. Moreover,

the occurrence locations of events may not be single points. Instead, they can occur along a path, or in an entire area. Moreover, the analysis for events that are associated with moving objects is also an interesting direction for future research.

- **Making QoS Evaluations Online.** Of course, the most challenging obstacle preventing the developed analytical framework being used online in the actual sensor nodes is its relatively high computation costs. The main purpose of developing the probabilistic models in this dissertation is to provide a framework for offline analysis. However, if the computation costs can be reduced without losing accuracy greatly, the analysis can conducted by the processors on the sensor nodes, and thus can be be utilized to make decisions more adaptively to the dynamic environment and network conditions. Whenever the task, the topology, or the channel quality changes, the network can simply re-calculate the optimal parameters or policies. Therefore, how to expedite the calculation without losing accuracy is a very important research topic.

Publications Resulted from This Dissertation

[1] Yunbo Wang, Mehmet C. Vuran, and Steve Goddard. Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. In *Proc. of IEEE RTSS 2009*, Washington, DC, Dec 2009.

[2] Yunbo Wang, Mehmet C. Vuran, and Steve Goddard. Stochastic analysis of energy consumption in wireless sensor networks. In *Proc. of IEEE SECON 2010*, Boston, MA, Jun 2010.

[3] Yunbo Wang, Mehmet C. Vuran, and Steve Goddard. Analysis of event detection delay in wireless sensor networks. In *Proc. of IEEE INFOCOM 2011*, Shanghai, China, Apr 2011.

[4] Yunbo Wang, Mehmet C. Vuran, and Steve Goddard. Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. *IEEE Trans. on Networking*, 20(1):305–318, Feb 2012.

Bibliography

[1] T.R. Abdelzaher, S. Prabh, and R. Kiran. On real-time capacity limits of multihop wireless sensor networks. In *Proc. of IEEE RTSS 2004*, pages 359–370, Dec 2004.

[2] Ö. B. Akan and I.F. Akyildiz. Event-to-sink reliable transport in wireless sensor networks. *IEEE/ACM Trans. on Networking*, 13(5):1016, 2005.

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

[4] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. A survey on wireless multimedia sensor networks. *Computer Networks*, 51(4):921–960, Mar 2007.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks Journal (Elsevier)*, 38(4):393–422, Mar 2002.

[6] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[7] G. Armitage. *Quality of service in IP networks: foundations for a multi-service Internet*. Macmillan Publishing Co., Inc. Indianapolis, IN, 2000.

[8] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, Mar 2000.

[9] N. Bisnik and A. Abouzeid. Queuing network models for delay analysis of multi-hop wireless ad hoc networks. In *IWCMC 2006: Proc. of the 2006 international conference on Wireless communications and mobile computing*, pages 773–778, Vancouver, British Columbia, Canada, 2006.

[10] N. Boughanmi and Y.Q. Song. A new routing metric for satisfying both energy and delay constraints in wireless sensor networks. *Journal of Signal Processing Systems*, 51(2):137–143, 2008.

[11] M Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proc. of ACM SenSys 2006*, Boulder, CO, Oct 2006.

[12] A. Burchard, J. Liebeherr, and S.D. Patek. A min-plus calculus for end-to-end statistical service guarantees. *IEEE Trans. on Information Theory*, 52(9):4105–4114, Sep 2006.

[13] Q. Cao, T. Yan, J. Stankovic, and T. Abdelzaher. Analysis of target detection performance for wireless sensor networks. In *Proc. DCOSS 2005*, pages 276–292, Marina del Rey, CA, Jul 2005.

[14] S. Chakrabarti and A. Mishra. QoS issues in ad hoc wireless networks. *IEEE Communications Magazine*, 39(2):142–148, 2002.

[15] D. Chen and P.K. Varshney. QoS support in wireless sensor networks: A survey.

In *International Conference on Wireless Networks*, pages 227–233. Citeseer, 2004.

[16] J. Chen, W. Xu, S. He, Y. Sun, P. Thulasiraman, and X. Shen. Utility-based asynchronous flow control algorithm for wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 28(7):1116–1126, Sept 2010.

[17] C.F. Chiasserini, R. Gaeta, M. Garetto, M. Gribaudo, D. Manini, and M. Sereno. Fluid models for large-scale wireless sensor networks. *Performance Evaluation*, 64(7-8):715–736, 2007.

[18] T.H. Cormen. *Introduction to algorithms*. The MIT press, 2001.

[19] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick. A Framework for QoS-based Routing in the Internet. *RFC2386, August*, 1998.

[20] R.L. Cruz. A calculus for network delay. i. network elements in isolation. *IEEE Trans. on Information Theory*, 37(1):114–131, Jan 1991.

[21] W. Dargie, X. Chao, and M.K. Denko. Modelling the energy cost of a fully operational wireless sensor network. *Telecommunication Systems*, 44(1-2):3–15, June 2010.

[22] I. Demirkol and C. Ersoy. Energy and delay optimized contention for wireless sensor networks. *Computer Networks*, 53(12):2106–2119, Aug 2009.

[23] I. Dietrich and F. Dressler. On the lifetime of wireless sensor networks. *ACM Trans. on Sensor Networks*, 5(1):1–39, Feb 2009.

[24] O. Dousse, C. Tavoularis, and P. Thiran. Delay of intrusion detection in wireless sensor networks. In *Proc. of ACM MobiHoc 2006*, page 165, Florence, Italy, May 2006.

[25] O. Dousse, P. Thiran, and M. Hasler. Connectivity in ad-hoc and hybrid networks. In *Proc. of IEEE INFOCOM 2002*, pages 1079–1088, New York, Jun 2002.

[26] E.J. Duarte-Melo and M. Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *Proc. of IEEE GLOBECOM 2002*, Taipei, Taiwan, Nov 2002.

[27] E.J. Duarte-Melo, M. Liu, and A. Misra. A modeling framework for computing lifetime and information capacity in wireless sensor networks. In *WiOpt 2004*, Cambridge, UK, Mar 2004.

[28] Enrique J. Duarte-melo and Mingyan Liu. Data-gathering wireless sensor networks: Organization and capacity. *Computer Networks*, 43:519–537, 2003.

[29] S. Eswaran, A. Misra, and T. La Porta. Utility-based adaptation in mission-oriented wireless sensor networks. In *Proc. of IEEE SECON 2008*, pages 278–286, San Francisco, CA, June 2008. IEEE.

[30] E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural. Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks. In *Proc. of IEEE INFOCOM 2005*, volume 4, pages 2646–2657 vol. 4, Miami, FL, Mar 2005.

[31] K.P. Ferentinos and T.A. Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Computer Networks*, 51(4):1031–1051, Mar 2007.

[32] M. Fidler. An end-to-end probabilistic network calculus with moment generating functions. In *Proc. of IEEE IWQoS*, pages 261–270, New Haven, CT, Jun 2006.

[33] M. Franceschetti, O. Dousse, D.N.C. Tse, and P. Thiran. Closing the gap in the capacity of wireless networks via percolation theory. *IEEE Trans. on Information Theory*, 53(3):1009–1018, 2007.

[34] R. G. Gallager. *Discrete Stochastic Processes*. Kluwer Academic Publishers, 1995.

[35] L. Galluccio and S. Palazzo. End-to-End delay and network lifetime analysis in a wireless sensor network performing data aggregation. In *Proc. of IEEE GLOBECOM 2009*, pages 146–151, Honolulu, HI, Nov 2009.

[36] K. Gopalan, T.-C. Chiueh, and Y.-J. Lin. Probabilistic delay guarantees using delay distribution measurement. In *Proc. of ACM MULTIMEDIA 2004*, pages 900–907, New York, NY, Oct 2004.

[37] M. Gribaudo, D. Manini, A. Nordio, and C.F. Chiasserini. Analysis of IEEE 802.15. 4 sensor networks for event detection. In *Proc. of IEEE Globecom 2009*, pages 152–157, Honolulu, Hawaii, Nov 2009.

[38] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J.A. Stankovic, T. Abdelzaher, and B. H. Krogh. Lightweight detection and classification for wireless sensor networks in realistic environments. In *Proc. of ACM SenSys 2005*, San Diego, CA, Nov 2005.

[39] V. C. Gungor, Ö. B. Akan, and I. F. Akyildiz. A real-time and reliable transport

(RT)^2 protocol for wireless sensor and actor networks. *IEEE/ACM Trans. on Networking*, 16(2):359–370, 2008.

[40] G. R. Gupta and N. B. Shroff. Delay analysis for multi-hop wireless networks. In *Proc. of IEEE INFOCOM 2009*, pages 412–421, Rio de Janeiro, Brazil, Apr 2009.

[41] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Trans. on Information Theory*, IT-46(2):388–404, Mar 2000.

[42] J. Haapola, Z. Shelby, C. Pomalaza-Raez, and P. Mahonen. Cross-layer energy analysis of multihop wireless sensor networks. In *Proc. of IEEE EWSN 2005*, Jan 2005.

[43] Holland Computing Center. Holland Computing Center - FireFly. Online: http://hcc.unl.edu/firefly/index.php.

[44] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, May 2001.

[45] IEEE. IEEE Std 802.15.4: IEEE standard for wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs), Oct 2003.

[46] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. on Networking*, 11(1):2–16, Feb 2003.

[47] T. Issariyakul and E. Hossain. Analysis of end-to-end performance in a multi-hop wireless network for different hop-level ARQ policies. In *Proc. of IEEE GLOBECOM 2004*, volume 5, pages 3022–3026 Vol.5, Dallas, TX, Nov 2004.

[48] D. Jung, T. Teixeira, and A. Savvides. Sensor node lifetime analysis: Models and tools. *ACM Trans. on Sensor Networks*, 5(1):1–33, Feb 2009.

[49] J. Kim, X. Lin, and N. Shroff. Optimal anycast technique for delay-sensitive energy-constrained asynchronous wireless sensor networks. In *Proc. of IEEE INFOCOM 2009*, pages 412–421, Rio de Janeiro, Brazil, Apr 2009.

[50] J. Kim, X. Lin, N.B. Shroff, and P. Sinha. Minimizing delay and maximizing lifetime for wireless sensor networks with anycast. *IEEE Trans. on Networking*, 18(2):515–528, Apr 2010.

[51] A. Koubaa, M. Alves, and E. Tovar. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks. In *Proc. of IEEE RTSS 2006*, pages 412–421, Rio de Janeiro, Brazil, Dec 2006.

[52] S. Kulkarni, A. Iyer, and C. Rosenberg. An address-light, integrated mac and routing protocol for wireless sensor networks. *IEEE/ACM Trans. on Networking*, 14(4):793–806, Aug 2006.

[53] V. G. Kulkarni. Fluid models for single buffer systems. *Frontiers in Queueing: Models and Applications in Science and Engineering*, pages 321–338, 1997.

[54] S. Kumar, A. Arora, and T.H. Lai. On the lifetime analysis of always-on wireless sensor network applications. In *Proc. of IEEE MASS 2005*, Washington, DC, Nov 2005.

[55] J.P. Lehoczky. Real-time queueing network theory. In *Proc. of IEEE RTSS 1997*, pages 58–67, San Francisco, CA, Dec 1997.

[56] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: accurate and scalable simulation of entire tinyos applications. In *Proc. of ACM SenSys 2003*, Los Angeles, CA, Nov 2003.

[57] H. Li, P. Shenoy, and K. Ramamritham. Scheduling messages with deadlines in multi-hop real-time sensor networks. In *Proc. of IEEE RTAS 2005*, pages 415–425, San Francisco, CA, Mar 2005.

[58] S. Liu, K.-W. Fan, and P. Sinha. CMAC: An energy efficient mac layer protocol using convergent packet forwarding for wireless sensor networks. In *Proc. of SECON 2007*, pages 11–20, San Diego, CA, Jun 2007.

[59] Y. Liu, F. Lo Presti, V. Misra, D. Towsley, and Y. Gu. Fluid models and solutions for large-scale IP networks. In *Proc. of ACM SIGMETRICS 2003*, pages 91–101, San Diego, CA, Jun 2003.

[60] D. Marco, E. J. Duarte-Melo, M. Liu, and D. L. Neuhoff. On the many-to-one transport capacity of a dense wireless sensor network and the compressibility of its data. In *Proc. IPSN'03*, Palo Alto, CA, Apr 2003.

[61] W.S. Marcus. An architecture for QoS analysis and experimentation. *IEEE/ACM Trans. on Networking (TON)*, 4(4):603, 1996.

[62] D. McDysan. *QoS and traffic management in IP and ATM networks*. McGraw-Hill, Inc. New York, NY, 1999.

[63] T. Melodia, M. C. Vuran, and D. Pompili. The state of the art in cross-layer design for wireless sensor networks. *Wireless Systems and Network Architectures in Next Generation Internet*, pages 78–92, 2006.

[64] P. Mohapatra, J. Li, and C. Gui. QoS in mobile ad hoc networks. *IEEE Wireless Communications*, 10(3):44–53, 2003.

[65] A. Muhammad Mahtab, B. Olivier, M. Daniel, A. Thomas, and S. Olivier. A Hybrid Model for Accurate Energy Analysis of WSN Nodes. *EURASIP Journal on Embedded Systems*, 2011, Jan 2011.

[66] Yoni Nazarathy and Gideon Weiss. The asymptotic variance rate of the output process of finite capacity birth-death queues. *Queueing Systems*, 59(2):135–156, June 2008.

[67] Randolph Nelson. *Probability, stochastic processes, and queueing theory: the mathematics of computer performance modeling*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[68] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: an Algorthmic Approach*. Dover Publications Inc., 1981.

[69] M. F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, 1989.

[70] M.F. Neuts, J. Guo, M. Zukerman, and H. L. Vu. The waiting time distribution for a TDMA model with a finite buffer and state-dependent service. *IEEE Trans. on Communications*, 53(9):1522–1533, Sep 2005.

[71] M. Noori and M. Ardakani. A probabilistic lifetime analysis for clustered wireless sensor networks. In *Proc. of IEEE WCNC 2008*, pages 2373–2378, Las Vegas, NV, 31 2008-April 3 2008.

[72] M. Noori and M. Ardakani. A probability model for lifetime of event-driven wireless sensor networks. In *SECON 2008*, pages 269–277, Jun 2008.

[73] M. Noori and M. Ardakani. Lifetime Analysis of Random Event-Driven Clustered Wireless Sensor Networks. *IEEE Trans. on Mobile Computing*, 10(10):1448–1458, Dec 2010.

[74] R.S. Oliver and G. Fohler. Probabilistic estimation of end-to-end path latency in wireless sensor networks. In *Proc. of IEEE MASS 2009*, pages 423 –431, Macau, China, Oct 2009.

[75] D. O'Neill, A. Goldsmith, and S. Boyd. Wireless network utility maximization. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1–8, San Diego, CA, Nov 2008. IEEE.

[76] W. Pak, J.-G. Choi, and S. Bahk. Tier based anycast to achieve maximum lifetime by duty cycle control in wireless sensor networks. In *Wireless Communications and Mobile Computing Conference, 2008. IWCMC 2008. International*, pages 123–128, Aug 2008.

[77] P. Park, P. Di Marco, P. Soldati, C. Fischione, and K.H. Johansson. A generalized markov chain model for effective analysis of slotted IEEE 802.15.4. In *Proc. of IEEE MASS 2009*, pages 130 –139, Macau, China, Oct 2009.

[78] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proc. of ACM SenSys 2004*, Baltimore, MA, Nov 2004.

[79] S. Pollin, M. Ergen, S.C. Ergen, B. Bougard, F. Catthoor, A. Bahai, and P. Varaiya. Performance analysis of slotted carrier sense ieee 802.15.4 acknowledged uplink transmissions. In *Proc. of IEEE WCNC 2008*, pages 1559–1564, Las Vegas, NV, Mar 2008.

[80] T. Sakurai and H.L. Vu. MAC access delay of IEEE 802.11 DCF. *IEEE Trans. on Wireless Communications*, 6(5):1702–1710, May 2007.

[81] J.B. Schmitt, F.A. Zdarsky, and L. Thiele. A comprehensive worst-case calculus for wireless sensor networks with in-network processing. In *RTSS 2007*, pages 193–202, Dec 2007.

[82] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Trans. on Mobile Computing*, 1(1):70–80, Jan 2002.

[83] R.C. Shah and J.M. Rabaey. Energy aware routing for low energy ad hoc sensor networks. In *Proc. of IEEE WCNC 2002*, Orlando, FL, Mar 2002.

[84] L. Shi and A.O. Fapojuwo. TDMA scheduling with optimized energy efficiency and minimum delay in clustered wireless sensor networks. *IEEE Trans. on Mobile Computing*, 9(7):927–940, July 2010.

[85] Y. Song, C. Zhang, and Y. Fang. Stochastic traffic engineering in multihop cognitive wireless mesh networks. *IEEE Trans. on Mobile Computing*, 9(3):305–316, Mar 2010.

[86] Y. Sun, O. Gurewitz, and D.B. Johnson. RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In *Proc. of ACM SenSys 2008*, Raleigh, NC, Nov 2008.

[87] O. Tickoo and B. Sikdar. Modeling queueing and channel access delay in unsaturated IEEE 802.11 random access MAC based wireless networks. *IEEE/ACM Trans. Networks*, 16(4):878–891, Aug 2008.

[88] MICA2 sensor node. web. http://www.xbow.com.

[89] NI USB-6210 multifunction DAQ. web. http://www.ni.com.

[90] TelosB sensor node. web. http://www.xbow.com.

[91] TinyOS. web. http://webs.cs.berkeley.edu/tos/.

[92] S. Toumpis and L. Tassiulas. Packetostatics: Deployment of massively dense sensor networks as an electrostatics problem. In *Proc. of IEEE INFOCOM 2005*, Miami, FL, Mar 2005.

[93] L. Van Hoesel, T. Nieberg, J. Wu, and P.J.M. Havinga. Prolonging the lifetime of wireless sensor networks by cross-layer interaction. *IEEE Wireless Communications*, page 79, 2004.

[94] S. Vural and E. Ekici. Probability distribution of multi-hop-distance in one-dimensional sensor networks. *Computer Networks*, 51(13):3727–3749, Sept 2007.

[95] M. C. Vuran and I. F. Akyildiz. XLP: A cross layer protocol for efficient communication in wireless sensor networks. *IEEE Trans. on Mobile Computing*, 9(11):1578–1591, Nov 2010.

[96] D. Wang, B. Xie, and D.P. Agrawal. Coverage and lifetime optimization of wireless sensor networks with gaussian distribution. *IEEE Trans. on Mobile Computing*, 7(12):1444–1458, Dec 2008.

[97] Q. Wang, M. Hempstead, and W. Yang. A realistic power consumption model for wireless sensor network devices. In *Proc. of IEEE SECON 2006*, pages 286–295, Reston, VA, Sep 2006.

[98] M. Xie and M. Haenggi. Towards an end-to-end delay analysis of wireless multihop networks. *Ad Hoc Networks*, 7(5):849 – 861, July 2009.

[99] Y. Xue, M. C. Vuran, and B. Ramamurthy. Cost-efficiency of anycast-based forwarding in duty-cycled wsns with lossy channel. In *Proc. of IEEE SECON 2010*, Boston, MA, Jun 2010.

[100] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Trans. Networks*, 12(3):493–506, June 2004.

[101] S.-N. Yeung and J. Lehoczky. End-to-end delay analysis for real-time networks. In *Proc. of IEEE RTSS 2001*, pages 299–309, London, UK, Dec 2001.

[102] Y. Yi and M. Chiang. Stochastic network utility maximisation—a tribute to Kelly's paper published in this journal a decade ago. *European Transactions on Telecommunications*, 19(4):421–442, Apr 2008.

[103] J. Yuan and W. Yu. Distributed cross-layer optimization of wireless sensor networks: A game theoretic approach. In *Proc. of IEEE GLOBECOM 2006*, pages 1–5, San Francisco, CA, Nov 2006. IEEE.

[104] Y.S. Yun and Y. Xia. Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications. *Mobile Computing, IEEE Transactions on*, 9(9):1308–1318, 2010.

[105] J. Zhang, G. Zhou, S.H. Son, J.A. Stankovic, and K. Whitehouse. Performance analysis of group based detection for sparse sensor networks. In *Proc. ICDCS 2008*, pages 111–122. IEEE, Jun 2008.

[106] M. Zorzi and R.R. Rao. Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance. *IEEE Trans. on Mobile Computing*, 2(4):337, 2003.

[107] M. Zúniga and Bhaskar Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans. on Sensor Networks*, 3(2), June 2007.