Network-Calculus-based Performance Analysis for Wireless Sensor Networks

Huimin She

Stockholm 2009

Thesis submitted to the Royal Institute of Technology in partial fulfillment of the requirements for the degree of Licentiate of Technology
She, Huimin

Network-Calculus-based Performance Analysis for Wireless Sensor Networks

TRITA-ICT/ECS AVH 09:03
ISSN 1653-6363
ISRN KTH/ICT/ECS/AVH-09/03-SE

Copyright © Huimin She, April 2009

Royal Institute of Technology (KTH)
School of Information and Communication Technology
S-164 40 Stockholm
SWEDEN
Abstract

Recently, wireless sensor network (WSN) has become a promising technology with a wide range of applications such as supply chain monitoring and environment surveillance. It is typically composed of multiple tiny devices equipped with limited sensing, computing and wireless communication capabilities. Design of such networks presents several technique challenges while dealing with various requirements and diverse constraints. Performance analysis techniques are required to provide insight on design parameters and system behaviors.

Based on network calculus, we present a deterministic analysis method for evaluating the worst-case delay and buffer cost of sensor networks. To this end, three general traffic flow operators are proposed and their delay and buffer bounds are derived. These operators can be used in combination to model any complex traffic flowing scenarios. Furthermore, the method integrates a variable duty cycle to allow the sensor nodes to operate at low rates thus saving power. In an attempt to balance traffic load and improve resource utilization and performance, traffic splitting mechanisms are introduced for mesh sensor networks. Based on network calculus, the delay and buffer bounds are derived in non-splitting and splitting scenarios. In addition, analysis of traffic splitting mechanisms are extended to sensor networks with general topologies. To provide reliable data delivery in sensor networks, retransmission has been adopted as one of the most popular schemes. We propose an analytical method to evaluate the maximum data transmission delay and energy consumption of two types of retransmission schemes: hop-by-hop retransmission and end-to-end retransmission.

We perform a case study of using sensor networks for a fresh food tracking system. Several experiments are carried out in the Omnet++ simulation environment. In order to validate the tightness of the two bounds obtained by the analysis method, the simulation results and analytical results are compared in the chain and mesh scenarios with various input traffic loads. From the results, we show that the analytic bounds are correct and tight. Therefore, network calculus is useful and accurate for performance analysis of wireless sensor network.

**Keywords:** Wireless sensor network, performance analysis, network calculus, traffic splitting
Acknowledgments

First I would like to express sincere gratitude to my supervisors Prof. Axel Jantsch, Prof. Li-Rong Zheng, and Dr. Zhonghai Lu. Prof. Jantsch and Prof. Zheng are very respectable persons for their knowledge and personality. Their understanding, encouragement and guidance have provided a good basis for the thesis. Dr. Lu helps me a lot during my research and study, even personal life. Many thanks go to him for his constructive comments and fruitful discussions.

I would also thank my co-supervisors Prof. Dian Zhou and Prof. Xuan Zeng at Fudan university for introducing me to the world of scientific research, and for their support and guidance.

Many thanks are due to current and former colleagues at ECS, especially in the iPack center, SAM and ESD groups, for their friendship and the pleasant atmosphere. There are too many names to be mentioned here, thank you all. I would also like to thank Dr. Ingo Sander, Docent Johnny Öberg, Prof. Ahmed Hemani, Prof. Elena Dubrova, Dr. Supriya Krishnamurthy and Dr. Adam Dunkels for their interesting lectures and discussions.

Also, special thanks should go to the administrative staff at ECS for their assistance in all kinds of issues, and the IT support group in ICT.

I am grateful to the Vinn Excellence centers program of Vinnova (The Swedish Governmental Agency for Innovation Systems). Without its financial support, this work could not have been possible.

I owe my loving thanks to my girlfriend Liping Wang for her understanding, continuous support and patience. I will always remember the days we spent together. In addition, I would like to thank my parents and brother for their unconditional support and encouragement.

Huimin She
Stockholm, April 2009
Contents

Acknowledgments v
List of Abbreviations xiii

1 Introduction 1
  1.1 Background ............................................. 1
  1.2 Motivation ............................................. 3
  1.3 Network Calculus Basis .................................. 4
  1.4 Contributions and Outline ................................. 7

2 Operator-based Performance Analysis 11
  2.1 Introduction ............................................. 11
      2.1.1 Wireless mesh sensor networks .................... 11
      2.1.2 Sensor network calculus ............................ 12
  2.2 System Model ............................................. 13
  2.3 Performance Analysis Based on Traffic Flow Operators 14
      2.3.1 Traffic passing ..................................... 15
      2.3.2 Traffic merging .................................... 16
      2.3.3 Traffic splitting ................................... 17
      2.3.4 The deterministic analysis method .................. 18
      2.3.5 An example ......................................... 18
  2.4 Energy Analysis .......................................... 20
  2.5 Numerical Results ........................................ 21
  2.6 Summary ................................................. 24

3 Traffic Splitting Mechanisms 25
  3.1 Introduction ............................................. 25
  3.2 System Model ............................................. 26
      3.2.1 Network topology ................................... 26
3.2.2 Traffic model ................................................. 26
3.3 Traffic Splitting in Mesh Sensor Networks .................. 27
  3.3.1 Non-traffic-splitting scenarios .......................... 28
  3.3.2 Traffic splitting scenarios .............................. 29
  3.3.3 End-to-end delay bound .................................. 32
  3.3.4 Numerical results ....................................... 32
3.4 Traffic Splitting for General Sensor Networks ............. 37
  3.4.1 Maximum delay ......................................... 38
  3.4.2 Maximum backlog ....................................... 38
3.5 Summary ..................................................... 39

4 Analysis of Retransmission Schemes 41
  4.1 Introduction ............................................... 41
  4.2 System Model .............................................. 42
  4.3 Analysis of Retransmission Schemes ....................... 43
    4.3.1 Hop-by-hop retransmission ............................ 43
    4.3.2 End-to-end retransmission ............................ 45
  4.4 Experiment Results ....................................... 46
    4.4.1 Experiment setup ...................................... 46
    4.4.2 Comparisons of two schemes ......................... 47
  4.5 Summary .................................................. 49

5 Case Study 51
  5.1 A Fresh Food Tracking System ............................. 51
  5.2 Experimental Evaluation .................................. 53
    5.2.1 Experiment setup ..................................... 53
    5.2.2 Simulation results for the chain scenario .......... 54
    5.2.3 Simulation results for the mesh scenario .......... 59
  5.3 Summary .................................................. 66

6 Conclusions and Future Work 67
  6.1 Conclusions ................................................ 67
  6.2 Future Work ................................................ 68

References 69
# List of Figures

1.1 A wireless sensor network for volcanic monitoring ........................................... 2
1.2 (a): arrive curve and service curve; (b): delay bound and backlog bound .......................................................... 4

2.1 Service Curve ......................................................................................................... 14
2.2 Traffic Flow Operators. (a) Traffic passing: one input flow and one output flow; (b) Traffic merging: multiple input flows and one output flow; (c) Traffic splitting: one input flow and multiple output flows. ......................................................................................... 15
2.3 An example of traffic flows .................................................................................... 19
2.4 An example of the sensor network .......................................................................... 21
2.5 Date delivery ratios with various duty cycles ......................................................... 22
2.6 Worst-case delays with various duty cycles ............................................................ 23
2.7 Minimum, average, and maximum backlog with different duty cycles .................. 24

3.1 (a) A mesh sensor network; (b) A relay node. ...................................................... 26
3.2 Routing in mesh sensor networks without splitting: (a) X-Y routing; (b) Y-X routing. ......................................................................................................................... 28
3.3 (a) Routing in mesh sensor networks with splitting; (b) Input and output flows of node (i,j). ................................................................................................................. 30
3.4 Maximum per-hop delay ......................................................................................... 33
3.5 Maximum backlog .................................................................................................... 34
3.6 Maximum, average, minimum per-hop delay ......................................................... 34
3.7 Maximum, average, minimum backlog ..................................................................... 35
3.8 Maximum end-to-end delay .................................................................................... 35
3.9 Maximum delay versus network size ....................................................................... 36
3.10 Traffic Splitting in General Sensor Networks ....................................................... 37
List of Figures

4.1 (a): Hop-by-hop retransmission; (b): End-to-end retransmission. .......................... 43
4.2 Compare the maximum transmission delay. .......................... 47
4.3 Compare the energy consumption. .......................... 48
4.4 Compare analytical maximum transmission delay with the simulation results. .......................... 48

5.1 A real-time fresh food tracking system .......................... 52
5.2 Chain scenario .......................... 54
5.3 Compare calculated delay with simulated delay .......................... 56
5.4 Compare calculated delay with maximum, mean, minimum simulated delay .......................... 57
5.5 Compare calculated delay with simulated delay .......................... 58
5.6 Compare calculated delay with maximum, mean, minimum simulated delay .......................... 59
5.7 Mesh scenario .......................... 60
5.8 Compare calculated delay with simulated delay (low traffic) .......................... 61
5.9 Compare calculated delay with simulated delay (heavy traffic) .......................... 62
5.10 Calculated backlog and simulated backlog (low traffic) .......................... 63
5.11 Comparison between calculated backlog and simulated backlog of selected nodes (low traffic) .......................... 64
5.12 Calculated backlog and simulated backlog of all nodes (heavy traffic) .......................... 65
5.13 Comparison between calculated backlog and simulated backlog of selected nodes (heavy traffic) .......................... 65
List of Tables

4.1 Experimental Parameters ........................................... 46

5.1 Parameters of the Chain Scenario ................................. 55
5.2 Parameters of the Mesh Scenario ................................. 60
List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>ACKnowledge</td>
</tr>
<tr>
<td>ARQ</td>
<td>Automatic Repeat reQuest</td>
</tr>
<tr>
<td>BER</td>
<td>Bit Error Rate</td>
</tr>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>ETS</td>
<td>Even Traffic Splitting mechanism</td>
</tr>
<tr>
<td>FIFO</td>
<td>First-In-First-Out</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile communications</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MAN</td>
<td>Metropolitan Area Network</td>
</tr>
<tr>
<td>NACK</td>
<td>Negative ACKnowledge</td>
</tr>
<tr>
<td>PAN</td>
<td>Personal Area Network</td>
</tr>
<tr>
<td>PSFQ</td>
<td>Pump Slowly and Fetch Quickly</td>
</tr>
<tr>
<td>PTS</td>
<td>Probabilistic Traffic Splitting mechanism</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency IDentification</td>
</tr>
<tr>
<td>RN</td>
<td>Relay Node</td>
</tr>
<tr>
<td>Rx</td>
<td>Receiving</td>
</tr>
<tr>
<td>SN</td>
<td>Source Node</td>
</tr>
<tr>
<td>TDMA</td>
<td>Time Division Multiple Access</td>
</tr>
<tr>
<td>Tx</td>
<td>Transmission</td>
</tr>
<tr>
<td>WFP</td>
<td>Wait for First Packet</td>
</tr>
<tr>
<td>WMN</td>
<td>Wireless Mesh Network</td>
</tr>
<tr>
<td>WMSN</td>
<td>Wireless Mesh Sensor Network</td>
</tr>
<tr>
<td>WSN</td>
<td>Wireless Sensor Network</td>
</tr>
<tr>
<td>WTS</td>
<td>Weighted Traffic Splitting mechanism</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This chapter provides a brief background and highlights design challenges in wireless sensor networks. We also give an overview of the research presented in the thesis and outline the author’s contributions.

1.1 Background

With the development of wireless communications and micro-electronics, wireless sensor network (WSN) has become a promising technology and received significant research attention [46, 2, 39, 15, 31, 19, 66, 43].

A typical sensor network consists of a large number of sensor nodes deployed either inside the phenomenon of interest or close to it. These sensor nodes are devices equipped with sensing, computation, and wireless communication capabilities. They take measurements and forward their observation values via the wireless interfaces to single or multiple fusion centers. The fusion center can also be called the sink node. Typical sensing tasks for sensors could be temperature, light, humidity, vibration, sound, etc. There are a variety of applications for WSNs, which typically involve in monitoring, tracking and controlling. Specific applications include environment surveillance, next-generation healthcare systems, structural monitoring, supply chain management, disaster area monitoring, and military assistance [24, 3, 19]. Figure 1.1 shows an application of sensor networks for volcanic monitoring [65]. When an earthquake or an eruption occurs, sensor nodes detect the seismic event and send data to a base station via a multi-hop network. Collecting and analyzing data from multiple base stations can produce precise mappings of the volcano.
Although WSNs share many commonalities with existing ad hoc networks, there are also a number of unique features and application requirements [3, 31, 19]. These features make the design of WSNs challenging.

- **Application specific** [7]: Sensor networks can be deployed in a variety of application scenarios. It is unlikely that there will be all-purpose solutions for all these potential possibilities. Therefore, it is essential to design a sensor network that can meet the requirements and constraints of specific tasks.

- **Constrained resource** [19]: Unlike traditional networks, sensor networks are limited in power, computational capability and communication bandwidth. New solutions are needed both to meet the requirements of a specific application and to consider the trade-offs between performance and cost. On the other hand, in order to allocate the limited resource, evaluating the performance of WSNs is therefore a crucial task.

- **Limited memory size** [50]: Too much memory space would increase the cost and size of sensors, while too little memory space can not meet the requirements of applications. Hence, it is essential to estimate the required memory spaces before the deployment of sensor networks.

- **Low power requirements** [47, 12]: Due to size and cost constraints, the tiny sensor nodes can only be equipped with a limited power source.
Moreover, in some application scenarios, recharge of power source is impossible. Hence, efficient power management takes on great importance in sensor networks. One of the challenges is to design power efficient protocols.

- **Unpredictability** [3, 2]: A sensor network is subject to a number of uncertainties from several factors. First, the sensor network may be deployed in an hostile environment dealing with uncontrollable aspects. Second, the wireless communication is unreliable due to interference, attenuation and fading effects. Third, some nodes may die over time and some new nodes may join later. As a consequence, the network topology and routing structures change dynamically.

## 1.2 Motivation

In many application scenarios of sensor networks, sensor data must be delivered to the base station within time constraints so that appropriate actions can be made. Hence, it is crucial to evaluate the performance limits, such as maximum delay, of WSNs in all conditions [26]. On the other hand, sensor networks present several technical challenges in terms of extremely low cost, low energy requirements and limited communication capabilities, while dealing with various workloads and diverse constraints [25, 48]. Addressing all these problems inevitably requires performance analysis techniques to provide insight on the design parameters and system behavior.

Since WSNs are in the early stage of development, most of the solutions are built, tested and evaluated either by simulation or testbeds. In [67], Woon et al. present a preliminary performance investigation, including throughput, packet delivery ratio, and average delay, of IEEE 802.15.4 standard focusing on multiple sources and multi-hop wireless sensor networks. In [23], the authors propose a mechanism to improve network coverage, increase throughput and reduce delay of a static sensor network by deploying several mobile sensors. In [71], Zhang et al. investigate the performance of an experimental wireless sensor network. The dynamic relaying and fixed relaying algorithms are tested and compared in terms of throughput, packet loss rate and data efficiency. Simulations are useful in evaluating the average performance for specific cases. However, they are usually time consuming. If one configuration parameter is changed, the simulation needs to be run again. Moreover, simulations are very difficult or even impossible to cover all worst cases. A more efficient and faster approach is therefore required.
to design and analyze a sensor network before it is deployed. The analysis needs to provide confidence that the network can meet its requirements and to indicate the performance of the network.

In this thesis, we have been orienting our research towards analytical performance evaluation for WSNs using network calculus, which is a useful theory for worst-case performance analysis in packet switching networks [17, 18, 9]. It has been developed for the worst-case analysis and resource dimensioning of WSNs by Schmitt et al. [51, 52, 53], which is also called sensor network calculus.

Next, we introduce basic terminologies of network calculus and summarize the author’s contributions.

1.3 Network Calculus Basis

Network calculus contains two sub classes, which are deterministic network calculus [17, 18, 9, 1] and stochastic network calculus [28]. In this thesis, network calculus refers to deterministic network calculus. Network calculus is a min-plus system theory for deterministic queuing systems [9]. It has been developed for deterministic queuing, allowing to derive deterministic guarantees on throughput and delay, and bounds on buffer sizes. Specifically, network calculus analyzes the worst-case behavior of a network rather than the average behavior.

Next some basic definitions and notations are provided. Detailed definitions and descriptions can be found in [9].
1.3. Network Calculus Basis

**Definition 1. Min-Plus Convolution**

Let \( f(t) \) and \( g(t) \) be wide-sense increasing functions \(^1\) defined over real number \( t \geq 0 \), and \( f(0) = g(0) = 0 \). Then their convolution under min-plus algebra is defined as:

\[
(f \otimes g)(t) = \inf_{0 \leq s \leq t} \{ f(t-s) + g(s) \} \quad (1.1)
\]

An arrival process can be modeled by its cumulative traffic \( R(t) \), which is defined as the number of bits that arrive in the time interval \([0, t]\). In particular, \( R(0) = 0 \) and \( R(t) \) is wide-sense increasing. Similarly, the output function \( R^*(t) \) of a system is the number of bits that depart from the system in time interval \([0, t]\). In particular, \( R^*(0) = 0 \) and \( R^*(t) \) is wide-sense increasing.

**Definition 2. Arrival curve**

Given a flow with input function \( R(t) \), a wide-sense increasing function \( \alpha(t) \) is an arrival curve for \( R(t) \) if and only if (Figure 1.2),

\[
\forall t \geq 0, s \geq 0 \text{ and } s \leq t : R(t) - R(t-s) \leq \alpha(s) \quad (1.2)
\]

One of the most commonly used arrival curve is the affine arrival curve \([9]\), which is defined as:

\[
\alpha(t) = \rho \cdot t + \sigma \quad (1.3)
\]

where \( \sigma \) and \( \rho \) represent the burst tolerance (in units of data) and the rate (in units of data per unit time), respectively. The average data rate \( \rho \) gives an indication of the expected traffic volume in a given period of time. And burstiness \( \sigma \) describes the maximum traffic above the average rate during any time interval.

**Definition 3. Service Curve**

Assume that a server process is able to process \( C(t) \) bits of input data until time \( t \). Then, a wide-sense increasing function \( \beta(t) \) is a service curve for \( C(t) \) if and only if (Figure 1.2),

\[
\forall t \geq 0, 0 \leq s \leq t : C(t) - C(t-s) \geq \beta(s) \quad (1.4)
\]

One of the most commonly used service curve is the rate-latency service curve \([9]\), which is defined as:

\[
\beta(t) = R[t-T]^+ \quad (1.5)
\]

\(^1\)A function \( h \) is wide-sense increasing if and only if \( h(s) \leq h(t) \) for all \( s \leq t \).
where $R$ represents the service data rate and $T$ represents the maximum service delay. If $x \geq 0$, notation $[x]^+ = x$; otherwise it equals to 0.

With the arrival curve and service curve, the following theorems can be derived based on the network calculus theory. The detailed descriptions and proofs of these theorems can be found in [9].

**Theorem 1. Delay bound**

Assume a traffic flow $R(t)$, constrained by arrival curve $\alpha(t)$, traverses a system that provides a service curve $\beta(t)$. At any time $t$, the virtual delay $D(t)$ satisfies,

$$D(t) \leq \sup_{t \geq 0} \{ \inf_{\tau \geq 0} \{ \tau : \alpha(t) \leq \beta(t + \tau) \} \} \quad (1.6)$$

The delay bound defines the maximum delay that would be experienced by a bit arriving at time $t$. Graphically, the delay bound is the maximum horizontal deviation between $\alpha(t)$ and $\beta(t)$ (Figure 1.2).

**Theorem 2. Backlog Bound**

Assume a traffic flow $R(t)$, constrained by arrival curve $\alpha(t)$, traverses a system that provides a service curve $\beta(t)$. The backlog $B(t)$ for all $t$ satisfies,

$$B(t) \leq \sup_{t \geq 0} \{ \alpha(t) - \beta(t) \} \quad (1.7)$$

The backlog is the amount of bits that are held inside the sensor node. The required buffer size of a sensor node is determined by the maximum backlog. Graphically, the backlog bound is the maximum vertical deviation between $\alpha(t)$ and $\beta(t)$ (Figure 1.2).

**Theorem 3. Output bound**

Assume a traffic flow $R(t)$, constrained by arrival curve $\alpha(t)$, traverses a system that provides a service curve $\beta(t)$. The output flow is constrained by the following arrival curve:

$$\alpha^*(t) = \sup_{s \geq 0} \{ \alpha(t + s) - \beta(s) \} \quad (1.8)$$
Theorem 4. Concatenation

Assume a flow sequentially traverses two systems which offer a service curve of $\beta_1$ and $\beta_2$, respectively. Then the concatenation of the two systems offers the flow the service curve $\beta(t)$, which is defined by,

$$
\beta(t) = (\beta_1 \otimes \beta_2)(t) = \inf_{0 \leq s \leq t} \{\beta_1(t - s) + \beta_2(s)\}
$$

(1.9)

where $\otimes$ represents min-plus convolution (Definition 1). If $\beta_1$ and $\beta_2$ are rate-latency service curves, i.e. $\beta_1(t) = R_1[t - T_1]^+$ and $\beta_2(t) = R_2[t - T_2]^+$, then $\beta_1 \otimes \beta_2 = R^*[t - T^*]^+$, where $R^* = \min(R_1, R_2)$ and $T^* = T_1 + T_2$.

Theorem 5. Aggregate Multiplexing

Consider a node multiplexing two flows 1 and 2 in First-In-First-Out (FIFO) order. Assume that the node provides a guaranteed service curve $\beta(t)$ to the aggregate of the two flows and flow 2 is constrained by an arrival curve $\alpha_2(t)$. Then, for any $\theta \geq 0$ and $t \geq \theta$, flow 1 is guaranteed the following service curve:

$$
\beta^\theta_1(t) = [\beta(t) - \alpha_2(t - \theta)]^+
$$

(1.10)

1.4 Contributions and Outline

The remainder of this thesis is structured as follows:

Chapter 2

In this chapter we first introduce the traffic model and service model of sensor networks. Then, we propose three general traffic flow operators, which are: traffic passing operator, traffic merging operator, and traffic splitting operator. Based on the traffic model and service model, we derive the delay bound and backlog bound of these operators. Finally, initial numerical results are given. Most of the material was published in:


Author’s contributions: The author developed the algorithm, implemented the experiments and wrote the manuscript.
Chapter 3

This chapter presents traffic splitting mechanisms in mesh sensor networks. We first introduce the system model, including network topology and traffic model. Then, a network calculus based method is proposed to analyze the maximum delay and backlog bound of non-splitting and splitting scenarios in mesh sensor networks. Numerical experiments are presented to compare the results for cluster-tree sensor networks and mesh sensor networks. In addition, we extend the traffic splitting mechanisms to sensor networks with generalized topologies.

Part of the results were published in:


Author’s contributions: The author developed the algorithms of deriving the two bounds, conducted the numerical experiments and wrote the manuscript.

An extended version of the conference paper was published in:


Author’s contributions: The author contributed with the algorithm, solution, experiments and wrote the manuscript.

Chapter 4

This chapter presents an analytical method to evaluate the maximum transmission delay and energy efficiency of two types of retransmission schemes, which are the hop-by-hop retransmission scheme and the end-to-end retransmission scheme. The results were published in:

1.4. Contributions and Outline

Author’s contributions: The author contributed with the idea, solution, experiments and wrote the manuscript.

Chapter 5

In this chapter, we present a case study on a sensor-network-based fresh food tracking system. Based on the case study, simulations are carried out using Omnet++. Simulation results on packet transmission delay and backlog are compared with the calculation results obtained from the network calculus analysis.

Chapter 6

This chapter summarizes the thesis and proposes several open topics for future work.

As described above, this thesis is based on the corresponding papers. The following paper is not included in the thesis:

Chapter 2

Operator-based Performance Analysis

In this chapter, we present a deterministic analysis method integrating three general traffic flow operators and a variable duty cycle for worst-case performance analysis in WSNs. The work in this chapter is mainly based on [57].

2.1 Introduction

2.1.1 Wireless mesh sensor networks

Wireless mesh network (WMN) is dynamically self-organized and self-configured, with the nodes in the network automatically establishing an ad hoc network and maintaining the mesh connectivity [5]. In recent years, it have received much attention due to its advantages, such as low cost, easy network maintenance, robustness, self-configuration, and reliable service coverage [4, 10, 33]. These features enable wireless mesh networks to be applied in many scenarios including broadband home networking, building automation, disaster recovery and environment surveillance. Moreover, WMNs can be integrated with other networks such as sensor networks, the Internet, and cellular networks, etc.. The concept of multihop mesh have been used in several wireless networking technologies, ranging from personal area networks (PANs) to local area networks (LANs) and textitmetropolitan area networks (MANs). Among these networks, one of the most popular
Chapter 2. Operator-based Performance Analysis

technologies is wireless sensor network which is normally based on IEEE standards 802.11s and 802.15.4.

A sensor network may contain a huge number of low-cost sensors that are densely deployed at some inspected sites. In these scenarios, the sensor network is likely to form a mesh structure which has significant potential for use in commercial applications [6]. Wireless mesh sensor networks (WMSN) merge advantages of WMNs and WSNs [62]. Firstly, they enable better overall connectivity than other topologies, such as cluster-tree topology. In cluster-tree sensor networks [32, 30], packets from the source nodes can only be transmitted to the sink through one path. Therefore, if one of the routers is broken, all of its children nodes can not communicate with the sink. Secondly, mesh networking supports path diversity, i.e., the source nodes can send packets to the sink through multiple paths. Path diversity enables better traffic load balance which is good for improving transmission reliability, mitigating congestion and balancing energy consumption. Thirdly, mesh networks can provide better service coverage. If one of the sensor nodes or links is faulty, alternative links can be quickly established from neighbor nodes or links. In [61], Tang et al. propose an architectural model of WMSNs which greatly extend the functionalities of traditional sensor networks.

2.1.2 Sensor network calculus

In real-time applications of WSNs like health care and intrusion detection, it is essential to ensure that the performance and cost are predictable under all conditions [54]. Since simulation based methods are time-consuming to find the worst-case performance, formal methods are required to dimension sensor networks in an analytical way rather than case-by-case simulations. In [13], Chiasserini et al. proposed a performance model based on Markov chains. They use this model to analytically investigate the trade-offs between performance characteristics (data delivery delay, network capacity) and the cost metric (energy consumption).

Recently, Schmitt et al. [51, 52, 53] extended network calculus to sensor network calculus, which can be used to derive delay bound and backlog bound in WSNs. In [32], Koubaa et al. proposed a methodology for the modeling and worst-case dimensioning of cluster-tree sensor networks. They provided a model of the cluster-tree topology characterized by its depth, the maximum number of child nodes and the maximum number of child routers per parent router. Based on this model, they derived “plug-and-play” expressions for the end-to-end delay bounds, buffering and bandwidth
requirement using network calculus. In [11], Cao et al. adopted a network calculus approach to investigate the power management problem in video sensor networks. In [54], Schmitt et al. incorporate in-network processing into worst-case calculus by taking into account computational resources on the sensor nodes. In [30], Jurcik et al. propose a method for worst-case dimensioning of cluster-tree WSNs with a mobile data sink.

2.2 System Model

We assume all the sensor nodes are stationary and there is only one fusion center, i.e., sink node. There are typically two types of traffic flows in WSNs. The first one is sensor reporting traffic which is from a sensor node to the sink. And the other type of traffic is from the sink to a sensor node. Generally, the traffic flowing towards the sensors is magnitudes lower than the traffic caused by the sensing events [51]. Furthermore, these two kinds of traffic can be transmitted using different radio channels so that they do not interfere. Therefore, our efforts concentrate on analyzing the sensor reporting traffic flows.

To characterize the traffic generated by the sensor nodes, we follow network calculus to model the arrival flow at a node using its cumulative traffic \( R(t) \). The cumulative traffic flow \( R(t) \) is constrained by an affine arrival curve \( \alpha(t) \), namely \( \alpha(t) = \rho \cdot t + \sigma \) (Definition 2 in Section 1.3), where \( \sigma \) and \( \rho \) represent the burst tolerance (in units of data) and the rate (in units of data per unit time), respectively. The average data rate \( \rho \) gives an indication of the expected traffic volume in a given period of time. And burstiness \( \sigma \) describes the maximum traffic above the average rate during any time interval.

In packet-switched networks, service curve has been abstracted to model the resource provided by the network [9]. It is abstracted to offer some guarantees to flows. We have introduced the definition of the general service curve in section 1.3. In WSNs, the service mainly depends on link layer characteristics, such as data transmission rate and packet scheduling. Moreover, in order to reduce energy consumption, low duty-cycle operations with a periodic sleeping and active process are always adopted in sensor networks [68, 69]. Only the nodes involving in transmitting or receiving are kept awake, while others stay in sleeping state. For simplicity, we assume that sleeping and wakeup processes of two communicating nodes are synchronized. These characteristics of the link layer can be modeled by the
rate-latency service curve,

$$\beta(t) = C \cdot \frac{S}{T_s} \cdot [t - (T_s - S)]^+$$ \hspace{1cm} (2.1)

where $C$ denotes the link capacity, $T_s - S$ denotes the duration of sleeping time, $S$ denotes the duration of the active time. So $S/T_s$ is the node’s duty cycle, which is defined as the percentage of time that the sensor node is active. An example of the rate-latency service curve is shown in Figure 2.1.

![Service Curve](image)

**Figure 2.1. Service Curve**

### 2.3 Performance Analysis Based on Traffic Flow Operators

There are several transmission schemes in WSNs, such as broadcast and unicast. These transmission schemes can generate many complex traffic flow scenarios. In order to model the traffic scenarios in a generalized way, we introduce three types of traffic flow operators, which are *traffic passing operator*, *traffic merging operator* and *traffic splitting operator* (Figure 2.2). These traffic flow operators can be used to describe all the combined traffic flow scenarios. Moreover, the performance analysis of a sensor network can be conducted through analysis of these operators.

In the rest of this section, we first derive the delay bound and backlog bound of the traffic operators separately based on network calculus. Then,
2.3. Performance Analysis Based on Traffic Flow Operators

![Traffic Flow Operators Diagram](image)

Figure 2.2. Traffic Flow Operators. (a) Traffic passing: one input flow and one output flow; (b) Traffic merging: multiple input flows and one output flow; (c) Traffic splitting: one input flow and multiple output flows.

A deterministic method for analyzing sensor networks is introduced. Afterwards, an example is presented.

2.3.1 Traffic passing

For the traffic passing operator, there is one input flow and one output flow (Figure 2.2). The input flow is constrained by the arrival curve defined in Equation (1.3). And the service provided by the sensor node is constrained by the service curve defined in Equation (2.1). Based on the theorems in section 1.3, the delay bound, backlog bound and output bound can be derived as follows.

The delay bound $D_{pa}$ is expressed as,

$$D_{pa} = \frac{\sigma T_s}{SC} + (T_s - S)$$  \hspace{1cm} (2.2)

The backlog bound $B_{pa}$ is derived by,

$$B_{pa} = \sigma + \rho(T_s - S)$$  \hspace{1cm} (2.3)

The output flow is constrained by the following arrival curve,

$$\alpha^*_{pa}(t) = \rho^* t + [\rho(T_s - S) + \sigma]$$  \hspace{1cm} (2.4)

where $\rho^* = \min(\rho, SC/T_s)$.

In the case that $SC/T_s < \rho$, the backlog will increase endlessly if the input traffic flows into the node continuously. To avoid this, measures should
be taken either to increase the service data rate or constrain the input data rate. Otherwise, packets will be dropped when the buffer is full. From the expression of the service curve, we can see that the service rate can be enhanced by increasing the duty cycle of the node. Furthermore, the input data rate can be adjusted by changing the node’s reporting rate.

2.3.2 Traffic merging

For the traffic merging operator, multiple traffic flows merge into one traffic flow at the sensor node, as shown in Figure 2.2. In this case, it is important that a service discipline should be applied to allocate the bandwidth. The service disciplines are used to control the order in which packets are served, and determine how packets from different connections interact with each other. In [72], Zhang described several service disciplines for packet-switching networks, for example, Delay Earliest-Due-Date, Virtual Clock, Fair Queuing. However, different service disciplines fit for different applications. The service discipline for sensor networks should be as simple as possible since the hardware resource in a sensor node is very limited. Therefore, we take the following two disciplines for bandwidth allocation in sensor networks. The first one is called rate-proportional allocation strategy, and the other one is called weight-proportional allocation strategy. In the rate-proportional allocation strategy, the allocated bandwidth is proportional to the data rate of each flow. This strategy can be applied in the case that all the network traffic has the same requirement on Quality of Service (QoS). While in weight-proportional allocation, each flow is assigned a weight value $w_i$, and the bandwidth is allocated according to the weight values. By choosing weight values, the node can provide differentiated service to different types of network traffic.

As shown in Figure 2.2, we assume that there are $N$ input flows, each of which is denoted by $F_i$. The output is an ensemble of traffic flows, each of which is denoted by $F_i^*$ corresponding to $F_i$. Let $C_i$ denote the bandwidth allocated to flow $F_i$. The arrival curve and service curve of flow $i$ are $\alpha_i(t) = \rho_i t + \sigma_i$ and $\beta_i(t) = \frac{SC_i}{T_i} [t - (T_i - S_i)]^+$, which are defined in Equation (1.3) and (2.1), respectively. For simplicity, we assume that the work period and active duration of all the sensor nodes are the same, i.e., $T_i = T_s$ and $S_i = S$. In this work, we adopt the rate-proportional allocation strategy, and $C_i$ can be calculated by,

$$C_i = \frac{\rho_i}{\sum_{j=1}^{N} \rho_j} \cdot \frac{SC}{T_s} \quad (2.5)$$
2.3. Performance Analysis Based on Traffic Flow Operators

It is straightforward to derive the expression for bandwidth allocation of the weight-proportional allocation strategy.

Based on the theorems in section 1.3), the delay bound, backlog bound and output bound can be derived by the following equations.

The delay bound \( (D_{me}^i) \) of flow \( i \) is,

\[
D_{me}^i = \frac{\sigma_i}{C_i} + (T_s - S) \quad (2.6)
\]

The backlog bound \( (B_{me}) \) of the node is derived by,

\[
B_{me} = \sum_{i=1}^{N} \sigma_i + (T_s - S) \sum_{i=1}^{N} \rho_i \quad (2.7)
\]

The output flow \( F^*_i \) is constrained by,

\[
\alpha_{me}^i(t) = C_i t + \sigma_i + \rho_i (T_s - S) \quad (2.8)
\]

2.3.3 Traffic splitting

In order to balance traffic load in the networks, an input flow may be split into multiple flows during its path to the destination. We define the traffic splitting operator (shown in Figure 2.2) to characterize this splitting process. Let \( \gamma_j \ (j = 1 \cdots M) \) and \( \sum_{j=1}^{M} \gamma_j = 1 \) denote the splitting factor, where \( M \) is the number of output paths. The arrival curve and service curve are defined in Equation (1.3) and (2.1), respectively. Then, the delay bound, backlog bound and output bound can be derived based on the theorems in section 1.3.

The delay bound \( (D_{sp}^j) \) is,

\[
D_{sp}^j = \frac{\gamma_j \sigma T_s}{SC} + (T_s - S) \quad (2.9)
\]

The backlog bound \( (B_{sp}) \) is,

\[
B_{sp} = \sigma + \rho (T_s - S) \quad (2.10)
\]

The output flow \( F^*_j \) is constrained by,

\[
\alpha_{sp}^j(t) = \min(\gamma_j \rho, SC/T_s) + \gamma_j [\sigma + (T_s - S)] \quad (2.11)
\]

For the traffic splitting operator, if \( \rho > MSC/T_s \), which means the output bandwidth can not satisfy the requirements, then packets will be dropped when the buffer is full. To avoid this, either the input data rate should be constrained or the service rate should be enhanced.
2.3.4 The deterministic analysis method

In this section, we present the deterministic analysis method as a whole. It works as follows:

- **Step 1.** Obtain the routing path of each traffic flow according to the topology of the sensor network and the routing algorithm.

- **Step 2.** Construct traffic flow scenarios based on the three traffic flow operators.

- **Step 3.** Compute the output flow, delay bound and backlog bound for each node starting from the source node.

- **Step 4.** Calculate the end-to-end delay bound.

In step 4, there are two ways to compute the end-to-end delay bound. In the first method, we compute the delay bounds of the aggregate input flows at each hop. And then the end-to-end delay bound is calculated as the sum of per-hop delays. This method is simple but a bit pessimistic in some cases, since the delay bound at each hop is contributed by the aggregation of all input flows. This method can be called per-hop based method, which has been used in [51].

The other method is to derive an equivalent service curve for an individual input flow based on the aggregate multiplexing theorem (Equation (1.10)) and the concatenation theorem (Equation (1.9)). Then the end-to-end delay bound is deduced using the arrival curve of the input flow and its equivalent service curve (Equation (1.6)). This method is called per-flow based method, which has been presented in [35, 32, 30].

2.3.5 An example

We show how the analysis method works through an example. As shown in Figure 2.3, assume that there are two input flows $F_1$ and $F_2$, which originates from node $a$ and $b$, respectively. The routing path of $F_1$ is $a \rightarrow c \rightarrow d \rightarrow e \rightarrow g$, and $F_2$ is $b \rightarrow c \rightarrow d \rightarrow f \rightarrow g$. We further assume that these two input flows are constrained by the affine arrival curves $\alpha_1 = \rho_1 t + \sigma_1$
and \( \alpha_2 = \rho_2 t + \sigma_2 \), i.e., \( F_1 \sim (\sigma_1, \rho_1) \) \(^1\) and \( F_2 \sim (\sigma_2, \rho_2) \), respectively. Let \( F_1^i \) and \( F_2^i \) \((i = a \cdots g)\) denote the corresponding output flow of \( F_1 \) and \( F_2 \) at node \( i \), respectively. Let the work period of all the nodes be \( T_s \), and the link capacity be \( C \). Let \( S_i \) \((i = a \cdots g)\) denote the active duration of node \( i \) in a period. Therefore, each node provides a service curve \( \beta_i = C \frac{S_i}{T_s} [t-(T_s-S_i)]^+ \) \((i = a \cdots g)\).

\[ \begin{align*}
\alpha_2 &= \rho_2 t + \sigma_2, \\
\alpha_1 &= \rho_1 t + \sigma_1.
\end{align*} \]

where \( \sigma_a = \sigma_1 + \rho_1 (T_s - S_a) \), \( \rho_a = \min(\rho_1, CS_a/T_s) \), \( \sigma_b = \sigma_2 + \rho_2 (T_s - S_b) \), and \( \rho_b = \min(\rho_2, CS_b/T_s) \).

Figure 2.3. An example of traffic flows

From Figure 2.3, we see that node \( a \) and \( b \) are traffic passing operators. So the delay bound \((D_1^a \text{ and } D_2^b)\), backlog bound \((B_a \text{ and } B_b)\), and output bound \((F_1^a \text{ and } F_2^b)\) can be derived accordingly.

\[ \begin{align*}
D_1^a &= \sigma_1 T_s/(S_a C) + (T_s - S_a) \\
D_2^b &= \sigma_2 T_s/(S_b C) + (T_s - S_b) \\
B_a &= \sigma_1 + \rho_1 (T_s - S_a) \\
B_b &= \sigma_2 + \rho_2 (T_s - S_b) \\
F_1^a &\sim (\sigma_a, \rho_a) \\
F_2^b &\sim (\sigma_b, \rho_b)
\end{align*} \]

(2.12)

\(^1\)In this thesis, \( F_1 \sim (\sigma_1, \rho_1) \) means that traffic flow \( F_1 \) is constrained by the affine arrival curve \( \rho_1 t + \sigma_1 \).
merging operator. The bandwidth allocation strategy is rate-proportional.

\[
D_1^c = \sigma_a (\frac{\rho_a + \rho_b}{\sigma_a C} T_s) + (T_s - S_c)
\]

\[
D_2^c = \sigma_b (\frac{\rho_a + \rho_b}{\sigma_a C} T_s) + (T_s - S_c)
\]

\[
B_c = \sigma_a + \sigma_b + (\frac{\rho_a + \rho_b}{\sigma_a C} T_s - S_c)
\]

\[
F_1^a \sim (\sigma_a + \rho_a (T_s - S_c), \min(\rho_a, \rho_1^c))
\]

\[
F_2^b \sim (\sigma_b + \rho_b (T_s - S_c), \min(\rho_b, \rho_2^c))
\]

(2.13)

where \(\rho_1^c = \frac{S_c C \rho_a}{(\rho_a + \rho_b) T_s}\) and \(\rho_2^c = \frac{S_c C \rho_b}{(\rho_a + \rho_b) T_s}\).

At node \(d\), there are one input link and two output links. All the calculation can follow the method for the traffic splitting operator. Using the same method, at node \(e\) and \(f\), the delay bound, backlog bound and output flow can be derived recursively according to the traffic passing operator.

After the above calculation, we can get the backlog bound at each node. The end-to-end delay bound can be calculated by either adding the individual delay at each node together or using the per-flow based method.

2.4 Energy Analysis

Since the communication energy consumption accounts for 70% of the total power in sensor networks [36], our work mainly focuses on analyzing the power used for communication rather than considering the power used for sensing and signal processing. We assume that the sleeping cycle of each sensor node is coordinated in a synchronized manner. After a sensor finishes its data transmission, it goes to sleep immediately. The energy consumption is contributed by four factors: data transmission, data reception, node sleeping, and radio startup. For simplicity, we assume that the power consumption of data transmission and reception are the same. Therefore, the energy consumption \((E_{\text{unit}})\) of a sensor node in a period is expressed as follows,

\[
E_{\text{unit}} = S \cdot P_{tx} + (T_s - S) \cdot P_{\text{sleep}} + E_{\text{start}}
\]

(2.14)

where \(E_{\text{start}}\) denotes the energy consumption for the radio switching from sleeping mode to active mode, \(P_{tx}\) denotes the power consumption for data transmission, and \(P_{\text{sleep}}\) denotes the power consumption when the node is in sleeping mode. This model provides an effective and simple way to calculate the energy consumption.
2.5 Numerical Results

We have implemented our deterministic analysis method in Matlab. The parameters used in the numerical experiments are as follows. A sensor network was generated by uniformly distributing 30 sensors in a $10 \times 10$ m area (Figure 2.4). The sensors are static after deployment. The sink node is located at the upper right corner. We adopt a table-driven routing protocol. According to Mica2 mote \(^2\) [16], the link capacity $C$ is 38.4 kbps and work period $T_s$ is 1.096 s [53]. We set the packet size 288 bits. The standard reporting frequency of each sensor is assumed to be 0.1 Hz, i.e. the sensor node sends one packet in every ten seconds, leading to a data rate of 28.8 bits/s. The burst size is assumed to be the amount of data generated in two seconds. In the experiments, the traffic load is changed by varying the reporting frequency from 0.1 Hz to 1 Hz. For simplicity, we assume that the Medium Access Control (MAC) protocol is ideal where there is no packet collision in the network.

\begin{figure}[h]
\centering
\includegraphics[width=\linewidth]{sensor_network.png}
\caption{An example of the sensor network}
\end{figure}

\(^2\)The Mica2 mote is a mote module used for low-power wireless sensor networks (see http://www.xbow.com)
Chapter 2. Operator-based Performance Analysis

Note that at a sensor node, its input data rate can be higher than its service rate due to a lower duty cycle configuration. If this happens, data loss may occur when the backlog buffer is full. To capture this in our experiments, we define data delivery ratio as the amount of data received by the sink versus that of data sent by the sources.

![Traffic load VS. Delivery ratio](image)

**Figure 2.5.** Date delivery ratios with various duty cycles

To study how duty cycle impacts the performance of sensor networks, we conduct several experiments. The number of sensors is 30. Figure 2.5 shows that the packet delivery ratio decreases as traffic load increases. The packet delivery ratio can be enhanced by increasing the duty cycle. In Figure 2.6, we can see that the end-to-end worst-case delay increases when the traffic load increases. With the same traffic load, the delay decreases as duty cycle increases. Therefore, when the data delivery ratio or the worst-case delay cannot meet the requirement of applications, the duty cycle needs to be increased. If the energy consumption needs to be decreased, lower duty cycle operations can be taken. Next, we study the energy consumption of the radio with different duty cycles. Following the parameters in [59], the
currents of the radio of Mica2 in the *Receiving* Rx mode and *Transmission* Tx (-5 dBm) mode are 7.0 mA and 7.1 mA, respectively. And the current in sleep mode is less than 1 µA [16]. The power supply is 3 V. Hence, the power consumption in active mode and sleep mode are approximately 21 mW and 3 µW. Based on Equation (2.14), we get the energy consumptions in a period are 0.47 mJ, 0.92 mJ, and 1.8 mJ when the duty cycles are 0.02, 0.04, and 0.08, respectively \(^3\).

![Traffic load VS. Worst-case delay](image)

**Figure 2.6.** Worst-case delays with various duty cycles

To show the backlog variation with duty cycles, we conduct the following experiment. The traffic load is 0.5 for both figures, and the duty cycle is 0.08. \(B_{min}\), \(B_{ave}\), and \(B_{max}\) denote the minimum backlog, average backlog, and maximum backlog, respectively. These values are obtained from backlog bounds at all sensor nodes. In Figure 2.7, we can see that the backlogs do not reduce much when duty cycle increases from 0.04 to 0.15. However, \(^3\)We assume that the start energy is ignored, since it does not impact the comparisons. Moreover, the start energy is typically very small (1 µJ) [47].
the average backlog is much smaller than the maximum backlog with the same duty cycle. In this example, adjusting duty cycle has smaller effect on reducing the backlogs.

![Backlog VS. duty cycle](image.png)

**Figure 2.7.** Minimum, average, and maximum backlog with different duty cycles

### 2.6 Summary

In this chapter, we defined three general traffic flow operators to model any traffic flow scenarios. And the delay and backlog bounds of these operators are separately derived. Based on the results of three operators, we proposed an operator-based deterministic performance analysis method integrating variable duty cycle for WSNs. With the numerical results, we show that increasing the duty cycle of sensors can improve data delivery ratio and reduce the packet transmission delay. However, increasing duty cycle can also increase energy consumption. Hence, different duty cycles should be determined according to different requirements and constraints.
Chapter 3

Traffic Splitting Mechanisms

In chapter 2, we present a deterministic method for performance analysis in sensor networks. In this chapter, this method is applied and extended to analyze the traffic splitting mechanisms. The work in this chapter is mainly based on [55] and [56].

3.1 Introduction

The traffic splitting mechanism plays an important role in traffic load balancing in packet-switched networks [34]. With traffic splitting, a traffic flow is divided into several sub-flows and each of them is sent to the destination through different routing paths. Traffic splitting is an effective strategy which permits increased throughput, enhanced transmission reliability and reduced network congestion. In [70], Zalesky et al. proposed an algorithm to split traffic across an optimal number of disjoint paths. It is shown that the spare capacity can be reduced and thus the overall performance of the system is improved.

For many applications of sensor networks, especially real-time applications, it is essential to ensure that messages are transmitted to the destinations before their deadlines. Moreover, it is important to ensure that messages which contain critical information are not dropped even in worst cases. In this chapter, we present an analytic method to evaluate traffic splitting mechanisms in mesh sensor networks.
3.2 System Model

3.2.1 Network topology

In the mesh sensor network we assume three kinds of sensor nodes and all the nodes are stationary. Their functions and properties are described as follows: 1) *Sink*: this node is responsible for gathering data from all the other sensors and sending the data to a base station. In our model, we assume that there is only one sink. 2) *Relay Node (RN)*: this kind of node has the ability to sense the events as well as forward messages for other nodes. 3) *Source Node (SN)*: this kind of node only has the ability to sense the events.

As shown in Figure 3.1(a), the mesh sensor network is composed of a single sink which is located at the center, and \((m \times n - 1)\) relay nodes, with \(m\) and \(n\) represent the number of nodes in horizontal and vertical dimension, respectively. For simplicity, we assume \(m\) and \(n\) are odd integers. However, it is straightforward to extend the analysis method when they are even integers. Let \(N_{i,j}\) denote the number of source nodes that the relay node \((i, j)\) connects.

![Figure 3.1](image)

*Figure 3.1.* (a) A mesh sensor network; (b) A relay node.

3.2.2 Traffic model

From previous descriptions, we know that relay nodes, source nodes and the sink have the ability to sense their environment and generate input
3.3 Traffic Splitting in Mesh Sensor Networks

traffic. We assume that all sensing data streams are uncorrelated. The input traffic of all source nodes is constrained by an affine arrival curve \( \alpha(t) = \rho t + \sigma \). Each relay node \((i, j)\) provides a guaranteed service constrained by a rate-latency service curve \( \beta_{i,j}(t) = R_{i,j}[t - T_{i,j}]^+ \). In particular, \( T_{i,j} \) does not include buffering latency due to backlog.

The input and out traffic of a relay node are shown in Figure 3.1(b). Let \( \alpha_{i,j}(t) \) and \( \alpha^*_{i,j}(t) \) denote the arrival curves of input flow and output flow, respectively. They are expressed by,

\[
\alpha_{i,j}(t) = (N_{i,j} + 1) \cdot \alpha(t) \quad (3.1)
\]

\[
\alpha^*_{i,j}(t) = \sup_{s \geq 0} \{ \alpha_{i,j}(t + s) - \beta_{i,j}(s) \} = \alpha_{i,j}(t) + \rho_{i,j} \cdot T_{i,j} \quad (3.2)
\]

From the definitions and theorems of network calculus, the maximum delay bound \( D_{i,j} \) and maximum required buffer bound \( B_{i,j} \) of relay node \((i, j)\) can be derived as follows:

\[
D_{i,j} = \frac{\sigma_{i,j}}{R_{i,j}} + T_{i,j} \quad (3.3)
\]

\[
B_{i,j} = \sigma_{i,j} + \rho_{i,j} T_{i,j} \quad (3.4)
\]

We assume \( R_{i,j} \geq \rho_{i,j} \), which means that the available bandwidth should be greater or equal to than the input data rate. Otherwise, the backlog will be increasing infinitely and thus packets will be dropped when the buffer is full. Moreover, the maximum delay bound may also become infinite.

3.3 Traffic Splitting in Mesh Sensor Networks

In this section, we derive the maximum delay bound and maximum required buffer bound for mesh sensor networks in two scenarios: without traffic splitting and with traffic splitting. We assume that the routing algorithm is a deterministic dimension order routing. In this routing policy, the \( m \times n \) mesh network is symmetric (Figure 3.2). Therefore, for an \( m \times n \) mesh, we only need to analyze part of the nodes with index \((i, j)\), where \( 1 \leq i \leq (m + 1)/2 \) and \( 1 \leq j \leq (n + 1)/2 \). For example, in a \( 5 \times 5 \) mesh network, only a \( 3 \times 3 \) mesh is needed to be analyzed (Figure 3.2, dashed frame).
3.3.1 Non-traffic-splitting scenarios

We assume that the packets from a relay node are transmitted to the sink through dimension order routing, i.e. either X-Y routing (Figure 3.2-(a)) or Y-X routing (Figure 3.2-(b)). We take the X-Y routing as an example. In this routing policy, the nodes in the same column (except the central column) can be considered the same, which means that the input flow, delay bound, backlog bound, and output flow of node \((1, j), (2, j), \ldots, (i, j), \ldots, ((m + 1)/2, j)\) are the same, where \(1 \leq j < (n + 1)/2\). Since the traffic pattern of the central column is different from other columns, we will analyze the central column separately. And for the central column, node \((i, (n + 1)/2)\) and node \((m + 1 - i, (n + 1)/2)\) can be considered the same.

![Figure 3.2. Routing in mesh sensor networks without splitting: (a) X-Y routing; (b) Y-X routing.](image)

According to the traffic model described in Section 3.2, the arrival curves of the input and output traffic flows of each relay node can be derived as follows. The bound conditions are defined as: \(\forall i = 0\) or \(j = 0\): \(\alpha_{i,j}^*(t) = 0\).

- If \(1 \leq i \leq \frac{m+1}{2}\) and \(1 \leq j < \frac{n+1}{2}\),

\[
\alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + \alpha_{i,j-1}^*(t) \quad (3.5)
\]
3.3. Traffic Splitting in Mesh Sensor Networks

- If \( 1 \leq i \leq \frac{m+1}{2} \) and \( j = \frac{n+1}{2} \),
  \[
  \alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + \alpha_{i-1,j}^*(t) + 2\alpha_{i,j-1}^*(t) \tag{3.6}
  \]
- If \( i = \frac{m+1}{2} \) and \( j = \frac{n+1}{2} \),
  \[
  \alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + 2[\alpha_{i-1,j}^*(t) + \alpha_{i,j-1}^*(t)] \tag{3.7}
  \]

Following Equation (3.2), the arrival curve of the output of relay node \((i,j)\) is derived as,
\[
\alpha_{i,j}^*(t) = \sup_{s \geq 0} \{\alpha_{i,j}(t + s) - \beta_{i,j}(s)\} = \alpha_{i,j}(t) + \rho_{i,j} \cdot T_{i,j} \tag{3.8}
\]

With Equations (3.5), (3.6), (3.7), (3.8), the arrival curves of the input and output traffic at each node can be recursively calculated. Then the maximum delay bound and maximum required buffer bound of each node can be calculated according to Equation (3.3) and (3.4), respectively. After getting the maximum delay at each node, the maximum delay bound of a traffic flow from the source to the destination can be calculated accordingly. The details are introduced in Section 3.3.3.

3.3.2 Traffic splitting scenarios

In the non-traffic-splitting scenario, a traffic flow is sent to the sink through a single routing path. This may result in traffic imbalance problem, since the links along the routing path are always occupied while others are vacant. For example, in Figure 3.2(a), the link between node \((1,1)\) and \((1,2)\) is always occupied, and the link between \((1,1)\) and \((2,1)\) is not used. In order to balance the traffic load and efficiently make use of the resources of sensor networks, three traffic splitting mechanisms are proposed, which are Even Traffic Splitting mechanism (ETS), Weighted Traffic Splitting mechanism (WTS), and Probabilistic Traffic Splitting mechanism (PTS). Along all routing paths between the source node and the sink, traffic flow is split at each relay node. As mentioned in the previous sections, the routing policy is minimal routing. Therefore, part of the packets are forwarded to the downstream node in the X direction (horizontal), and the other part in the Y direction (vertical).

The traffic model is the same as that described in Section 3.2.2. We assume that the traffic outputted at node \((i,j)\) will be routed along the X
direction with probability $p_{i,j}$ and the Y direction with probability $(1 - p_{i,j})$, where $0 \leq (p_{i,j}) \leq 1$. An illustration of the input and output traffic at node $(i, j)$ is shown in Figure 3.3(b). Based on Equation (3.5) and (3.8), the arrival curves of input and output traffic of each node can be recursively calculated by the following equations. We define the bound conditions as: $\forall i = 0$ or $j = 0$: $\alpha_{i,j}^*(t) = 0$ and $p_{i,j} = 0$.

- If $1 \leq i < \frac{m+1}{2}$, $1 \leq j < \frac{n+1}{2}$,

$$\alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + p_{i,j-1}\alpha_{i,j-1}^*(t) + (1 - p_{i-1,j})\alpha_{i-1,j}^*(t) \quad (3.9)$$

- If $1 \leq i < \frac{m+1}{2}$ and $j = \frac{n+1}{2}$,

$$\alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + 2p_{i,j-1}\alpha_{i,j-1}^*(t) + \alpha_{i-1,j}^*(t) \quad (3.10)$$

- If $i = \frac{m+1}{2}$ and $1 \leq j < \frac{n+1}{2}$,

$$\alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + \alpha_{i,j-1}^*(t) + 2(1 - p_{i-1,j})\alpha_{i-1,j}^*(t) \quad (3.11)$$

Figure 3.3. (a) Routing in mesh sensor networks with splitting; (b) Input and output flows of node $(i,j)$. 
3.3. Traffic Splitting in Mesh Sensor Networks

- If \( i = \frac{m+1}{2} \) and \( j = \frac{n+1}{2} \),

\[
\alpha_{i,j}(t) = (N_{i,j} + 1)\alpha(t) + 2\alpha_{i,j-1}^*(t) + 2\alpha_{i-1,j}^*(t)
\] (3.12)

From the arrival curve of the input and the service curve of each node, the maximum delay bound and maximum required buffer bound of each node can be calculated according to Equation (3.3) and (3.4), respectively. Based on the individual delay at each node, the maximum delay bound of a traffic flow from the source to the destination can be calculated accordingly. The details are introduced in Section 3.3.3.

By assigning different kinds of values to \( p_{i,j} \), three traffic splitting mechanisms for mesh sensor networks are introduced:

- **Even Traffic Splitting mechanism (ETS):** As shown in Figure 3.3-(a), in even traffic splitting mechanism, traffic is evenly split at each node. That is to say, 50% of packets flow to downstream nodes in the X direction and 50% of packets flow to downstream nodes in the Y direction. In this case, the probability \( p_{i,j} \) equals to 0.5. Therefore, the arrival curve of input and output traffic, the maximum delay bound and the maximum required buffer bound can be derived accordingly.

- **Weighted Traffic Splitting mechanism (WTS):** In weighted traffic splitting mechanism, traffic is split at each node not evenly. The packets outputted at each node will be routed along the X direction with probability \( p \), and the Y direction with probability \( 1 - p \), where \( 0 \leq p \leq 1 \). For simplicity, we assume that the values of \( p_{i,j} \) of each node are the same in this work. However, it is straightforward to extend it so that each node can set its own splitting coefficient \( p_{i,j} \). The value can be adjusted according to the requirements and constraints of practical applications. By setting \( p \), we can derive the expressions of input and output arrival curve recursively. Then the maximum delay bound and the maximum required buffer bound can be derived accordingly. In fact, ETS can be regarded as a special case of WTS when \( p = 0.5 \).

- **Probabilistic Traffic Splitting mechanism (PTS):** When a node receives a packet to be routed to downstream nodes, it has to determine which downstream node the packet should be forwarded to. In probabilistic traffic splitting mechanism, the relay node \((i,j)\) generates a random number \( p_{i,j} \) \( (0 \leq p_{i,j} \leq 1) \). Depending on different
requirements and constraints, this random number can be generated according to different distributions, such as uniform distribution and exponential distribution. Then the packet is forwarded to the downstream nodes with the probability $p_{i,j}$ and $1 - p_{i,j}$ in the X direction and the Y direction, respectively. Using the same method, the maximum delay bound and the maximum required buffer bound can be derived accordingly.

### 3.3.3 End-to-end delay bound

The end-to-end delay bound of a traffic flow can be derived through two approaches\(^1\). The maximum end-to-end delay of the whole network is equal to the delay bound of a traffic flow along the longest path in the network.

For example, in Figure 3.3(a), the maximum delay may happen between node $(1, 1)$ and the sink, so using the first approach the maximum delay can be calculated by $D = D_{1,1} + D_{1,2} + D_{1,3} + D_{2,3} + D_{3,3}$, where $D_{i,j}$ denotes the individual maximum delay of node $(i, j)$.

While using the second approach, the maximum end-to-end delay of an individual flow can be derived as following: we assume that at node $(1, 1)$ there are input flow 1 and 2, which are constrained by arrival curve $\alpha_1$ and $\alpha_2$, respectively. And node $(1, 1)$ provides a service curve $\beta_{1,1}$ to the aggregation of input flow. For flow 1, its guaranteed service curve ($\beta_1^\theta$) provided by node $(1, 1)$ can be derived based on Equation (1.10), namely, $\beta_1^\theta(t) = [\beta_{1,1}(t) - \alpha_2(t - \theta)]^+$, where $\theta \geq 0$ and $t \geq \theta$. Similarly, the guaranteed service curve for flow 1 provided by other nodes along its routing path can be derived. Then, an end-to-end equivalent service curve for flow 1 can be deduced based on Equation (1.9). With the arrival curve and equivalent service curve, the end-to-end delay of flow 1 can be derived using Equation (1.6).

### 3.3.4 Numerical results

We conduct several numerical experiments to illustrate the performance of traffic splitting mechanisms in mesh sensor networks. We assume that the size of the mesh network is $5 \times 5$, and the number of source nodes connect to each relay node is 2. Therefore, the network consists of one sink, 24 relay nodes, and 50 source nodes.

\(^1\)We have introduced these two approaches in Section 2.3.4. Please refer to it for details.
3.3. Traffic Splitting in Mesh Sensor Networks

The average input data rate $\rho$ is set to 15.36 bits/s, which roughly corresponds to sending a packet every 20 seconds. The burstiness $\sigma = 40$ bits. The Mica-2 motes [16] are assumed to be the sensor nodes, with maximum data forwarding rate 19.2 kbps. If the sensors are operated with duty cycle 11.5%, the maximum data forwarding rate $R$ is 2488 bits/s, and latency $T$ is 96 ms [52]. Therefore, each relay node provides a rate-latency service curve $\beta(t) = R[t - T]^+$, where $R = 2488$ bits/s, and $T = 0.096$ s. To study the delay bound and backlog bound of the mesh sensor networks, we choose a routing path with 5 hops from the source to the sink.

In the figures, NTS denotes Non-Traffic Splitting; ETS denotes Even Traffic Splitting; WTS denotes Weighted Traffic Splitting; and PTS denotes Probabilistic Traffic Splitting. In WTS, the probability $p$ is 0.6. In PTS, the random number $p_{i,j}$ is uniformly distributed.

![Comparison of maximum per-hop delays](image)

**Figure 3.4.** Maximum per-hop delay

The delay and backlog bounds at each relay node along a 5-hop path are shown in Figure 3.4 and Figure 3.5, respectively. The figures show that the per-hop delay bound and backlog bound are becoming bigger if the relay node is closer to the sink. This is mainly due to the fact that all the traffic is accumulated at the sink in sensor networks. The two bounds under traffic
splitting mechanisms are lower than those without traffic splitting.

![Comparison of maximum backlogs](image)

**Figure 3.5.** Maximum backlog

![Per-hop delay](image)

**Figure 3.6.** Maximum, average, minimum per-hop delay
Figure 3.7. Maximum, average, minimum backlog

Figure 3.8. Maximum end-to-end delay

Figure 3.6 shows the maximum, average, and minimum per-hop delay bounds of the relay nodes. Compared with NTS, the maximum per-hop
delay bound in ETS, WTS, and PTS, is reduced by 37.7%, 38.4%, and 45.2%, respectively. The minimum per-hop delay bounds are the same, which is 0.15 s. The average delay bound in the three traffic splitting mechanisms is reduced by 16.1% on average, compared with that in NTS. For the backlog bound, similar conclusions can be drawn from Figure 3.7. In a word, the reduction of maximum delay bounds and maximum backlog bounds has significant effect on improving the performance of sensor networks.

The end-to-end delay bounds from the source to sink are shown in Figure 3.8, which also reveals that using traffic splitting mechanisms will reduce the end-to-end delay bound and thus improve the performance of the network. The end-to-end delay bounds scaling with network size are shown in Figure 3.9 (The number of nodes in mesh sensor network are 16, 36, 64, and 144 respectively). This figure shows that adopting traffic splitting mechanisms will improve scaling properties. In conclusion, by splitting traffic among diverse routing paths in mesh sensor networks, the end-to-end delay can be decreased and the network resource utilization can be improved.
3.4 Traffic Splitting for General Sensor Networks

In this section, we present a method to analyze the performance of traffic splitting mechanisms for sensor networks with general topologies. Basically, traffic splitting process consists of two steps: establishing multiple routing paths and allocating a certain amount of traffic on each path according to some rules. There are plenty of literatures on how to setup multiple routing paths in ad hoc networks [40, 22, 20]. Our work in this thesis does not focus on finding multipath routes but on analyzing the performance of different traffic splitting mechanisms. Therefore, we assume that multiple paths have already been established between a source and a destination node. We then analyze the performance of different traffic splitting schemes in terms of maximum transmission delay and maximum required buffer size.

We assume that there are \( n \) disjoint paths available for data transmission between a source and a destination node (Figure 3.10). 'Disjoint' means that any two paths have no nodes in common except the source node and destination node. We assume that a source node generates a packet of size \( lM \) bits every \( T_M \) seconds. The packet is split into \( M \) fragments of size \( k \) \((k > l)^2\) bits. The source node then transmits these fragments over \( n \) paths with allocation coefficients \( m_i, i = 1, \ldots, n \), i.e., \( m_i \) fragments are transmitted over path \( i \). These allocation parameters can be generated according to predefined policies. For example, in [21], Djukic and Valaee presented an optimization algorithm to find an optimal allocation which can maximize the network lifetime under reliability and energy constraints. There are \( m_i \) fragments of size \( k \) transmitting over path \( i \) every \( T_M \) seconds.

![Figure 3.10. Traffic Splitting in General Sensor Networks](image)

\footnote{During the splitting process, overhead is added to each fragment. So the total size of fragments after splitting is larger than the original packet.}
We can use the following affine arrival curve to constrain this traffic scenario,
\[
\alpha_i(t) = \frac{km_i}{T_M} \cdot t + \left\lceil \frac{(k+1)M}{n} \right\rceil
\] (3.13)

where \( km_i/T_M \) (in bits/s) is the average data rate, and \( \left\lceil \frac{(k+1)M}{n} \right\rceil \) (in bits) is the burstiness which describes the maximum traffic above the average data rate during any time interval. \([x]\) is defined as the minimum integer greater than or equal to \( x \). We use the service model introduced in Section 3.2.2 to model the resource provided by the sensor nodes, i.e. the node provides a rate-latency service curve \( \beta(t) = R[t-T]^+ \), where \( R \) denotes the guaranteed average service rate, and \( T \) is the maximum latency of the service.

Let \( N_i \) denote the number of nodes in path \( i \) and \( \beta_i^j = R_i^j[t - T_i^j]^+ \) \((i = 1, \ldots, n; j = 1, \ldots, N_i)\) denote the service curve provided by node \( j \) at path \( i \). Based on the concatenation theorem (see Equation (1.9)), an equivalent service curve \( \beta_i \) for path \( i \) can be derived as follows,
\[
\beta_i = \beta_i^1 \otimes \beta_i^2 \otimes \cdots \otimes \beta_i^{N_i} = R_i^{\text{max}}[t - T_i^{\text{sum}}]^+
\] (3.14)

where \( R_i^{\text{max}} = \max(R_1^i, R_2^i, \ldots, R_{N_i}^i) \) and \( T_i^{\text{sum}} = T_1^i + T_2^i + \cdots + T_{N_i}^i \).

### 3.4.1 Maximum delay

After we get the equivalent service curve, the maximum transmission delay \( D_i \) of any fragment on path \( i \) can be calculated based on the delay bound theorem (see Equation (1.6)),
\[
D_i = \sup_{t \geq 0} \left\{ \inf_{\tau \geq 0} \{ \tau : \alpha_i(t) \leq \beta_i(t + \tau) \} \right\}
\]
\[
= \left\lceil \frac{(k+1)M}{n} \right\rceil + T_i^{\text{sum}}
\] (3.15)

### 3.4.2 Maximum backlog

We show how to derive the maximum backlog of each node. For path \( i \), the input arrival curve at the source is shown in Equation (3.13), i.e. \( \alpha_1^i(t) = \alpha_i(t) \), and the service curve provided by node \( j \) \((j = 1, \ldots, N_i)\) is \( \beta_j^i = R_j^i[t - T_j^i]^+ \). According to the connection relations, the input arrival curve of current node is the output arrival curve of its previous node. Based
on output bound theorem (see Equation (1.8), the input arrival curve of every node in path $i$ can be recursively derived by the following equation,

$$
\alpha_{i}^{j+1}(t) = \alpha_{i}^{j*}(t) \\
= \sup_{s \geq 0} \{ \alpha_{i}^{j}(t + s) - \beta_{i}^{j}(s) \} \\
= \alpha_{i}^{j}(t) + \rho_{i}^{j} \cdot T_{i}^{j} \tag{3.16}
$$

where $j = 1, \cdots, N_i$, and $\rho_{i}^{j}$ denotes the average data rate of the input traffic of node $j$ on path $i$.

After we derive the input arrival curve, the maximum backlog $B_{i}^{j}$ of node $j$ on path $i$ can be calculated based on the backlog bound theorem (see Equ. (1.7)),

$$
B_{i}^{j} = \sup_{s \geq 0} \{ \alpha(s) - \beta(s) \} \\
= \sigma_{i}^{j} + \rho_{i}^{j} \cdot T_{i}^{j} \tag{3.17}
$$

where $\sigma_{i}^{j}$ denotes the burstiness of the input traffic of node $j$ on path $i$.

### 3.5 Summary

In this chapter, we first present a system model of mesh sensor networks. Based on this model, the maximum delay bound and maximum required buffer bound are derived using network calculus for analyzing non-traffic-splitting and splitting mechanisms. We introduce three traffic splitting mechanisms, which are even traffic splitting (ETS), weighted traffic splitting (WTS), and probabilistic traffic splitting (PTS). In the numerical experiments, the maximum delay bound and maximum required buffer bound are compared in different scenarios: non-splitting case, the three splitting mechanisms, cluster-tree topology without splitting. From the results, we show that the delay bound and buffer requirement bound are reduced while applying those traffic splitting mechanisms in mesh sensor networks. Furthermore, in our mesh sensor network model, both the delay and buffer requirement bounds are lower than those in cluster-tree sensor networks.
Chapter 4

Analysis of Retransmission Schemes

This chapter presents an analytical evaluation of two retransmission schemes: hop-by-hop retransmission and end-to-end retransmission. The work in this chapter is mainly based on [58].

4.1 Introduction

Data transmission in wireless sensor networks is unreliable due to several factors such as fading, shadowing and multipath effects of radio propagation. One of the most common approaches for enhancing transmission reliability is retransmission [45] [64] [44]. Park et al. [45] propose a scalable framework for reliable downstream data delivery using a Wait-for-First-Packet (WFP) pulse. In [64], Wan et al. propose a reliable transport protocol called PSFQ (Pump Slowly and Fetch Quickly). These two protocols are typical examples that make use of hop-by-hop retransmissions. In [44], Pai et al. present an adaptive retransmission mechanism which allows a fusion center to select the sensors to retransmit their local information according to the reliability of the received information. This protocol belongs to end-to-end retransmission.

Many previous works have been done on reliable transport issues in experimental ways, however, there still lack of analytical techniques to evaluate different reliable transport solutions. In [37], Liu et al. analyze the roles of packet retransmission and erasure coding in the reliable transport of WSNs by establishing the probability models. In this paper, we propose analytical
techniques to evaluate retransmission schemes in WSNs. We first introduce
the traffic model, service model and energy model. Based on these models
and network calculus, we analytically evaluate the maximum packet trans-
fer delay and energy efficiency of two basic retransmission schemes, which
are hop-by-hop retransmission and end-to-end retransmission. From the
experiment results, the maximum delay and energy consumption of these
two schemes are compared in several scenarios. Moreover, the analytical
maximum delay is compared with the simulation results. With our method,
appropriate retransmission scheme can be chosen based on different require-
ments and constraints.

4.2 System Model

To characterize the traffic generated by the sensor nodes, we follow net-
work calculus to model the arrival flow at a node using its cumulative traffic
$R(t)$. We use an affine arrival curve $\alpha(t)$ to constrain the cumulative traffic
flow, namely, $\alpha(t) = \rho \cdot t + \sigma$ (Definition. 2 in Section 1.3). To model the
processing capability of a node, we use the same service curve as Equation
2.1 (Section 2.2), i.e.,

$$\beta(t) = C \cdot \frac{S}{T_s} \cdot [t - (T_s - S)]^+ \quad (4.1)$$

Following the energy model presented in [47], we abstract the energy
consumption of a packet transmission between two nodes in a similar way,

$$E = 2E_{\text{start}} + \frac{L}{R}\left(P_{tx} + P_{rx} + 2P_{cir}\right) \quad (4.2)$$

where $E_{\text{start}}$ represents the energy for startup the radio; $P_{tx}$ and $P_{rx}$ repre-
sent the power consumption of the radio in transmission mode and receive
mode, respectively; $P_{cir}$ represents the power consumption of the electronic
circuitry; $L$ denotes the packet length in bits; and $R$ denotes the transmis-
sion data rate. The energy consumption in the sleeping mode is ignored
since it is much smaller than that for packet transmission or reception [47].
However, it is straightforward to extend our model to include the energy
consumption in the sleep mode.
4.3 Analysis of Retransmission Schemes

There have been a lot of papers on designing retransmission schemes in WSNs [45] [44] [60]. These retransmission schemes can be classified into two basic categories, namely hop-by-hop retransmission and end-to-end retransmission (Figure 4.1).

We assume that there is a multi-hop path with \( n \) hops between a source node \( S \) and a destination node \( D \). And there is an automatic repeat request (ARQ) mechanism running until a packet successfully arrives at the receiver. A packet is not accepted as long as any bit of the packet is received with error (for non-coded systems). Furthermore, we assume an ideal MAC protocol where there is no interference and collision, so packet delivery failures are only due to channel errors. The packet error rate \( p_e \) can be computed by \( p_e = 1 - (1 - p_b)^L \), where \( p_b \) is bit error rate (BER).

4.3.1 Hop-by-hop retransmission

In hop-by-hop retransmission scheme, at every hop, the receiver checks the correctness of the packet and requests for a retransmission with an negative-acknowledgment (NACK) packet until a correct packet arrives. After that, an acknowledgment (ACK) packet is sent to the transmitter indicating a successful transmission. An example is shown in Figure 4.1. If the first packet transmission fails between A and B, then B sends an NACK packet to A asking for a retransmission. After that, A retransmits the packet. B sends an ACK packet after successfully receiving the packet.
Let $m_i$ denote the number of transmission trials at hop $i$, and $p_i$ denote the packet error rate at hop $i$. Then, the transmission delay and energy consumption can be derived as follows.

We assume the length of an ACK and NACK packet is denoted by $L_a$. At the source node $S$, the arrival curve is expressed by $\alpha_1(t) = p_1 \cdot t + \sigma_1$. According to (4.1), the service curve at hop $i$ ($1 \leq i \leq n$) is expressed as,

$$\beta_i(t) = C \cdot \frac{S_i}{T_s} \cdot [t - (T_s - S_i)]^+$$

(4.3)

where $S_i$ denotes length of the slot assigned to link $i$. Since the input of current hop equals the output of previous hop, i.e. $\alpha_i(t) = \alpha_{i-1}^*(t)$ ($2 \leq i \leq n$), the arrival curve of the traffic at the $i^{th}$ ($1 \leq i \leq n$) hop can be recursively derived based on Theorem 3 (Section 1.3, Equation (1.8)),

$$\alpha_i^*(t) = \sup_{s \geq 0} \{\alpha_i(t + s) - \beta_i(s)\} = \alpha_i(t) + p_i \cdot (T_s - S_i)$$

(4.4)

Based on Theorem 1 (Section 1.3, Equation (1.6)), (4.3) and (4.4), the maximum delay at hop $i$ can be derived as,

$$D_i = \sup_{t \geq 0} \{\inf_{\tau \geq 0} \{\tau : \alpha_i(t) \leq \beta_i(t + \tau)\}\} = \frac{\sigma_i T_s}{C S_i} + (T_s - S_i)$$

(4.5)

At each hop, the expected number of transmissions can be evaluated by $1/(1 - p_i)$. Therefore, the expected maximum delay ($D_{hbb}$) of sending a packet from $S$ to $D$ can be calculated by summing up the delays at each hop,

$$D_{hbb} = \sum_{i=1}^{n} \frac{1}{1 - p_i} D_i$$

(4.6)

The energy consumption is contributed by two factors: data packets and ACK (NACK) packets. For simplicity, the energy consumption for decoding is ignored although it is straightforward to include it. According to the energy model, the energy consumption at the $i^{th}$ hop can be calculated by,

$$E_i = 2E_{\text{start}}^i + \frac{L + L_a}{R} (P_{tx}^i + P_{rx}^i + 2P_{cir}^i)$$

(4.7)

Therefore, the total expected energy consumption $E_{hbb}$ of transmitting a packet from $S$ to $D$ can be computed by,

$$E_{hbb} = \sum_{i=1}^{n} \frac{1}{1 - p_i} E_i$$

(4.8)
4.3. Analysis of Retransmission Schemes

4.3.2 End-to-end retransmission

In end-to-end retransmission scheme, the intermediate nodes simply forward received packets to the next hop and do not check the correctness of the packets. When a packet arrives at the destination $D$, $D$ checks the packet, and asks for a retransmission with an NACK packet directly to $S$ if the packet is incorrect. Otherwise, it sends an ACK packet to $S$ indicating a successful packet transmission. In this scheme, ACKs/NACKs can be sent to source nodes directly. This is reasonable since the sink node (i.e. the destination) is usually very powerful and it can reach every node in the network. See example in Figure 4.1.

Let $p_i$ denote the packet error rate at hop $i$, and $m$ denote the number of transmission trials. Then, the transmission delay and energy consumption can be derived as follows.

In this scheme, the retransmission is performed in an end-to-end manner, so we can derive an equivalent service curve for the whole link based on Theorem 4 (Section 1.3, Equation (1.9)) and Equation (4.3),

$$\beta_{e2e} = \beta_1 \otimes \beta_2 \otimes \cdots \otimes \beta_n = R_{e2e} \cdot (t - T_{e2e})$$

(4.9)

where $R_{e2e}$ and $T_{e2e}$ can be calculated by,

$$R_{e2e} = \min_{1 \leq i \leq n} (C \cdot \frac{S_i}{T_s}), \quad T_{e2e} = \sum_{i=1}^{n} (T_s - S_i)$$

(4.10)

According to the traffic model, the arrival curve of the input flow at $S$ is defined as: $\alpha_{in}(t) = \rho_{in} \cdot t + \sigma_{in}$. Based on Theorem 1 (Section 1.3, Equation (1.6)), the maximum delay $D_{st}$ for one single transmission from $S$ to $D$ can be calculated by,

$$D_{st} = \sup_{t \geq 0} \{ \inf_{\tau \geq 0} \{ \tau : \alpha_{in}(t) \leq \beta_{e2e}(t + \tau) \} \} = \frac{\sigma_{in}}{R_{e2e}} + T_{e2e}$$

(4.11)

In end-to-end retransmission, the total expected number of transmissions can be evaluated by $1/p_{st}$, where $p_{st} = \prod_{i=1}^{n} (1 - p_i)$. Then, the expected maximum delay $D_{e2e}$ can be calculated by,

$$D_{e2e} = \frac{1}{p_{st}} D_{st}$$

(4.12)

In the end-to-end retransmission scheme, only the sink node needs to send ACK and NACK packets, other intermediate nodes simply forward
data packets. According to the energy model, the energy consumption at the $i^{th}$ hop can be calculated by,

$$E_i = 2E_{\text{start}}^i + \frac{L}{R}(P_{tx}^i + P_{rx}^i + 2P_{\text{cir}}^i)$$

(4.13)

Therefore, the total expected energy consumption $E_{\text{hhh}}$ of transmitting a packet from $S$ to $D$ can be computed by,

$$E_{e2e} = \frac{1}{p_{st}} \left[ \sum_{i=1}^{n} E_i + \frac{L_a}{R} (P_{tx}^i + P_{rx}^i + 2P_{\text{cir}}^i) \right]$$

(4.14)

In (4.14), the first item computes the energy consumption for transmitting data packets, while the second item computes the energy for ACK and NACK packets transmissions.

### 4.4 Experiment Results

#### 4.4.1 Experiment setup

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Notation</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tx power</td>
<td>$P_{tx}$</td>
<td>19.1</td>
<td>mW</td>
</tr>
<tr>
<td>Rx power</td>
<td>$P_{rx}$</td>
<td>14.6</td>
<td>mW</td>
</tr>
<tr>
<td>Circuit power</td>
<td>$P_{\text{cir}}$</td>
<td>12</td>
<td>mW</td>
</tr>
<tr>
<td>Start energy</td>
<td>$E_{\text{start}}$</td>
<td>1.0</td>
<td>uJ</td>
</tr>
<tr>
<td>Data pkt length</td>
<td>$L$</td>
<td>240</td>
<td>bits</td>
</tr>
<tr>
<td>ACK(NACK) pkt length</td>
<td>$L_a$</td>
<td>80</td>
<td>bits</td>
</tr>
<tr>
<td>Link capacity</td>
<td>$C$</td>
<td>19.2</td>
<td>kbps</td>
</tr>
</tbody>
</table>

In this section, the maximum transmission delay and energy consumption of hop-by-hop and end-to-end retransmission schemes are compared. The parameters used in experiments are shown in Table 4.1, which follow those used in [47] [16]. The link distance is randomly selected between 5 m and 10 m, which is typical for most applications. We set the frame length $T_s$ and slot length $S$ to 0.2 s and 0.01 s, respectively. The input data rate of end-to-end retransmission scheme $\rho_{in} = 30$ bps, which corresponds to one packet in every eight seconds. The burstiness is set to 240 bits, which
4.4. Experiment Results

equals to the size of one packet. For hop-by-hop scheme, the number of ACK (NACK) packets are the same as data packets, so the data rate at the first hop $\rho_1 = (1 + L/L_a)\rho_{in}$.

4.4.2 Comparisons of two schemes

Figure 4.2 shows the comparisons of the maximum delay with required target success probability varying. From this figure, we can see that the maximum delays of both schemes increase as the target success probability increases. Further more, when the BER is low, i.e., $(1e - 4)$, the maximum delay of end-to-end scheme is less than that of hop-by-hop scheme. But when the BER is high, i.e., $(1e - 3)$, the hop-by-hop scheme has less delay. This indicates that when the BER is high, more trials of retransmissions are required by end-to-end scheme to achieve the same target success probability.

![Comparison of Maximum Delay](image)

Figure 4.2. Compare the maximum transmission delay

Figure 4.3 plots the energy consumption varies with the required target success probability. In this figure, we observe the end-to-end scheme consumes more energy than the hop-by-hop scheme both for low and high BERs. Especially, when the BER is $(1e - 3)$, the energy consumption of the end-to-end scheme is much higher than the other one. The reason is that, in the end-to-end scheme, erroneous packets are not dropped until their destination is reached. And the energy for delivering these packets is wasted.
When BER is higher, there are more erroneous packets which would lead to higher energy consumption.

Figure 4.3. Compare the energy consumption

Figure 4.4. Compare analytical maximum transmission delay with the simulation results
4.5. Summary

To validate results of delay bound, we compare the analytical results with the simulation results in a chain scenario. The simulations are performed using Omnet++ 3.3. The path length is 4 hops and BER is $5e^{-4}$. Other parameters are shown in table 4.1. From Figure 4.4, we observe that all the simulation values are within the scopes of the analytical results. This indicates network calculus performs well on bounding the packet transfer delay. For end-to-end and hop-by-hop retransmission scheme, the analytical delays are 4.3% and 5.8% bigger than simulated maximum delays, respectively.

4.5 Summary

Due to the unreliable wireless links and limited energy budget, providing reliable data transmission has turned out to be a non-trivial problem in wireless sensor network. Retransmission has been adopted as one of the most prevalent schemes for addressing this issue. In this work, we presented analytical techniques to evaluate the maximum transmission delay and energy consumption of two categories of retransmission schemes: hop-by-hop retransmission and end-to-end retransmission.

With the experiment results, the maximum packet transfer delay and energy efficiency of two types of retransmission schemes are studied and compared. Moreover, our analytical method for deriving delay bound was validated through simulations.
Chapter 5

Case Study

In this chapter, we conduct a case study on a sensor network based fresh food tracking system. Using Omnet++, experiments are carried out to compare the analytical results with the simulation results.

5.1 A Fresh Food Tracking System

Many types of food products such as fruits, vegetables and meat have a limited freshness period before they spoil or lose nutritional value. Thus their shelf life can be greatly affected by the storage conditions during the transportation and storage process. A report from Billerud [8] revealed that, in the European market, approximately 10% of the whole cargo of fruits and vegetables coming from different parts of the world is deteriorated during the transportation process because of inappropriate storage conditions, and thus leads to a loss of billions of dollars per year. Among the factors that accelerate food’s rot, improper temperature and humidity conditions are the most important ones. Therefore, efficient and real-time tracking of these two factors is an essential point to be addressed [41].

There have been a variety of technologies for fresh food tracking [27]. Early Auto-ID technology can record the time and date information of food products using barcodes. A retailer can pay awareness and sort the products according to their expiry dates. However, barcodes have disadvantages like inability to be scanned when dirty or scrapped, and impossible to be updated. The next generation solution - Radio-Frequency IDentification (RFID) - fixes these problems and is becoming more and more popular. However, both barcode and RFID based technologies have their limitations.
They are normally used to record time information of food products. They cannot be used to monitor or detect the products’ storage conditions (such as temperature and humidity) which have great impacts on their freshness periods. For example, oranges can be preserved for several weeks in an appropriate environment. However, if the temperature or humidity ratio of the storage room is too high, their freshness periods can be greatly shortened. In order to prolong the freshness periods of food products and thus reduce cost, it is important to provide an appropriate temperature and humidity environment for the products during their transportation and storage processes.

In the following, we introduce a practical application of sensor networks for real-time tracking the freshness of food products. This system can be deployed at two locations: refrigerated food trucks and refrigerated warehouses. In this thesis, we describe the system deployed in a refrigerated food truck as an example.

![Diagram of a real-time fresh food tracking system](image)

**Figure 5.1.** A real-time fresh food tracking system

In the scenario of real-time fresh food (e.g. meat, vegetable, fruits) tracking, tags integrating temperature and humidity sensors are scattered
5.2 Experimental Evaluation

in food bins which are filled with fruits and vegetables in a refrigerated truck carriage (Figure 5.1). In order to save energy, the sensors work in two modes: active and sleep. In the active mode the node is fully working and is able to transmit and receive data, while in the sleep mode it can not take part in the network activity.

These sensors are responsible for collecting the corresponding information of food. All the data collected by sensors are sent to a base station, which may be placed on the top of the truck. The base station then transmits the data to a remote server through GSM (Global System for Mobile communications) or GPRS (General Packet Radio Service) networks. Thus a customer at the remote server side can read and analyze the data in real-time. If something is wrong or abnormal actions have happened, he can send instructions to the base station to take measures, such as lowering the temperature of the cooling system or sprinkling water onto fresh vegetables and fruits, to protect the food from deteriorating. In addition, there is a wired connection between the base station and the driver monitor. So the driver can also read the information collected by the network and take proper measures if necessary. The size of sensor networks applied in this application depends on the size of trucks. For small trucks, a 2-hop or 3-hop sensor network is enough, while for large trucks, a network of more hops is needed.

5.2 Experimental Evaluation

In order to validate the network calculus based method, we conduct experimental evaluations to compare the results obtained by the analytical method with the measured results. To carry out the evaluation in a controllable setting, we conduct experiments in a simulation environment rather than in a field trial. A simulation environment allows us to create a very realistic wireless sensor network behavior while still controllable. The simulation is implemented using Omnet++ version 3.3. It is based on the case study described in Section 5.1.

5.2.1 Experiment setup

The network consists of one sink node and multiple sensor nodes. A simple Time Division Multiple Access (TDMA) based medium access control protocol is used in the network. The protocol is based on ideas in [63] and [49]. As we described in chapter 2, let $T$ denote the duration of a TDMA
cycle. The cycle is divided into multiple time slots. The number of time slots should be not less than the number of sensor nodes. Each sensor node is assigned one time slot; the remaining time slots are used for contention-based access. For simplicity, we assume that two communicating nodes work in a synchronized manner. With this synchronized TDMA, packets can be transmitted without collisions since a time slot is allocated to exactly one node.

At the startup of the network, the sink node initializes the network by broadcasting a synchronization message in its time slot. This message contains TDMA cycle and slot allocation information and other control information. This process is repeated until every node received the message. Thereafter, all the nodes are synchronized with the TDMA cycle and thus they can start transmitting messages to the sink without collisions. In order to maintain time synchronization, the sink node should periodically broadcast a synchronization message. This MAC protocol is simple and efficient for small scenarios, such as the fresh food tracking system.

The link capacity follows the standard settings of the Chipcon CC2420 [14] Radio Frequency (RF) transceiver. The raw data rate of CC2420 is 250 kbps. However, this can never be achieved in any practical application because of effects like interference and channel sharing. From our measurements using the Tmote Sky platform [42], this value is around 150 kbps. Therefore, we set the link capacity to 150 kbps.

We use two scenarios in our simulations: the chain scenario and the mesh scenario.

5.2.2 Simulation results for the chain scenario

Figure 5.2.

Figure 5.2 shows an example of a chain scenario. All nodes are equally spaced in a straight line, and neighboring nodes are placed 2 m apart. In the simulation, the chain contains 10 sensor nodes and one sink. The sink

---

1The CC2420 is a single-chip 2.4 GHZ IEEE 802.15.4 compliant RF transceiver for low-power wireless applications [14].
5.2. Experimental Evaluation

Table 5.1. Parameters of the Chain Scenario

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Notation</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet size</td>
<td>$L$</td>
<td>240</td>
<td>bits</td>
</tr>
<tr>
<td>Link capacity</td>
<td>$C$</td>
<td>150</td>
<td>kbps</td>
</tr>
<tr>
<td>TDMA cycle</td>
<td>$T_c$</td>
<td>120</td>
<td>ms</td>
</tr>
<tr>
<td>Time slot</td>
<td>$S_c$</td>
<td>8</td>
<td>ms</td>
</tr>
</tbody>
</table>

is located at the right end. The path length from a sensor node to the sink varies from 1 hop to 10 hops. This scenario can correspond to the application of a food tracking system in a refrigerated warehouse or truck. The parameters used in simulations are shown in Table 5.1. All traffic in the chain is from a sensor node to the sink, generated as follows: at a periodic interval, every node (except the sink) creates a packet and sends it towards the sink in its time slot. Since a node might relay packets from other node, it is assumed that the packets are queued in a FIFO manner.

We consider two kinds of input traffic loads, which are low traffic load and heavy traffic load. In the low input traffic case, each sensor node generates a packet every ten seconds. So the average input data rate is 24 bps. We assume the burst tolerance is equal to 240 bits, which corresponds to the size of one packet. Hence, each sensory data traffic flow is bounded by the arrival curve $\alpha_l(t) = 24 \cdot t + 240$. In the heavy input traffic case, each sensor node generates one packet every second, which leads to an average input data rate of 240 bps. The burst tolerance is also equal to the size of one packet. As a consequence, each sensory data traffic flow is bounded by the arrival curve $\alpha_h(t) = 240 \cdot t + 240$. According to Table 5.1 and Equation (2.1), the average service rate of one time slot is $R_{cts} = C \cdot S_c / T_c = 10$ kbps, and the service latency is $T_{cts} = T_c - S_c = 0.112$ s. Therefore, each node provides a service curve $\beta_c(t) = R_{cts}[t - T_{cts}]^+$. With the arrival curves and service curves, the end-to-end maximum delay can be derived based on the method presented in Section 2.3.4. We use the per-flow based method to derive the end-to-end maximum delay. Next, we compare the calculated delays with the delays obtained from simulations.

Figure 5.3 shows, for the chain scenario, how the actual packet transmission delay varies with the path length. Moreover, the actual delay is compared with the calculated delay. The entire simulation runs until every node has sent 200 packets. In the figure, the dashed line denotes the worst-
Chapter 5. Case Study

In Figure 5.3, for 3-hop paths, the calculated worst-case delay is 6.2% bigger than the maximum simulated delay. For 5-hop, 7-hop and 9-hop paths, the calculated worst-case delays are 9.0%, 12.4%, and 13.9% bigger than those obtained from simulations, respectively.

Figure 5.4 shows the comparison between calculated worst-case delay and simulated maximum, mean, minimum delay. The path length varies from 2 hops to 10 hops. And the differences between the simulated maximum delay and the calculated worst-case delay are 4.8%, 6.2%, 8.8%, 9.0%, 11.9%, 12.4%, 16.9%, 13.9%, 14.2%. From this figure, we can see that the calculated worst-case delays are very close to the simulated maximum delays when the
5.2. Experimental Evaluation

Figure 5.4. Compare calculated delay with maximum, mean, minimum simulated delay

path lengths are small (i.e. less than 6 hops). While the path lengths become bigger, the differences between the calculated worst-case delays and the simulated maximum delays also become larger in general. This indicates that the probability of a ‘real’ worst-case delay appearing in simulations is very small if the distance (i.e. number of hops) from the source to the sink is big. In these cases, much more simulations are needed to find the real worst-case delay.

In the following simulation, the input traffic load is heavy, i.e., a sensor node generates a packet every second. The entire simulation runs until every node has sent 200 packets. In Figure 5.5, the dashed line denotes the worst-case delay obtained from the network calculus based method, and the solid line denotes the delay of packets transmission obtained from simulations. From Figure 5.5, we can see that the actual of transmission delays of most packets are very close to the calculated worst-case delays. For 3-hop, 5-hop, 7-hop and 9-hop paths, the calculated worst-case delays are 4.1%, 6.5%, 9.3%, and 10.4% above the simulated maximum delays, respectively.
Figure 5.5. Compare calculated delay with simulated delay

Figure 5.6 shows the comparison between calculated worst-case delay and simulated maximum, mean, minimum delay. The path length varies from 2 hops to 10 hops. And the differences between the simulated maximum delay and the calculated worst-case delay are 3.2%, 4.1%, 5.5%, 6.5%, 6.9%, 9.3%, 8.5%, 10.4%, 9.6%. From this figure, we can see that the calculated worst-case delays are very close to the simulated maximum delays when the path lengths are small (i.e. less than 6 hops). While the path lengths become bigger, the differences also become larger. Furthermore, compared with the results of the low input traffic case, the differences between the calculated worst-case delays and the simulated maximum delays of the heavy input traffic case are generally smaller. This indicates that the worst-case delays
5.2. Experimental Evaluation

Figure 5.6. Compare calculated delay with maximum, mean, minimum simulated delay

can be more easily captured by simulations if the input traffic load is heavier.

5.2.3 Simulation results for the mesh scenario

Figure 5.7 shows an example of an $n$ by $m$ mesh scenario. This scenario can correspond to a food tracking system in a refrigerated truck or warehouse with sensors manually deployed. In our simulation, the size of the mesh is 4 by 4, and the sink is located at the up-right corner. The routing method is minimal dimension order routing, i.e., packets from a source node are sent to the sink through dimension order via a minimal-hop path. The parameters used in simulations are shown in Table 5.2. All traffic in the chain is from a sensor node to the sink, generated as follows: at a periodic interval, every node (except the sink) creates a packet and sends it towards the sink in its
time slot. Since a node might relay packets from other node, it is assumed that the packets are queued in a FIFO manner.

![Mesh scenario diagram]

Figure 5.7. Mesh scenario

Table 5.2. Parameters of the Mesh Scenario

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Notation</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet size</td>
<td>$L$</td>
<td>240</td>
<td>bits</td>
</tr>
<tr>
<td>Link capacity</td>
<td>$C$</td>
<td>150</td>
<td>kbps</td>
</tr>
<tr>
<td>TDMA cycle</td>
<td>$T_m$</td>
<td>160</td>
<td>ms</td>
</tr>
<tr>
<td>Time slot</td>
<td>$S_m$</td>
<td>8</td>
<td>ms</td>
</tr>
</tbody>
</table>

In the following simulations, we adopt two kinds of input traffic scenarios: low traffic scenario and heavy traffic scenario. In these two traffic scenarios, a sensor node generates a packet every ten seconds and every second, respectively. So the average input data rates are 24 bps and 240 bps. We assume the burst tolerance is equal to 240 bits, which corresponds to the size of one packet. Hence, the sensory data traffic flows are bounded by the arrival curves $\alpha_l(t) = 24 \cdot t + 240$ and $\alpha_h(t) = 240 \cdot t + 240$. According to Table 5.2 and Equation (2.1), the average service rate of one time slot is $R_{mts} = C \cdot S_m/T_m = 7.5 \text{ kbps}$, and the service latency is $T_{mts} = T_m - S_m = 0.152 \text{ s}$. Therefore, each node provides a service curve
\( \beta_m(t) = R_{mts}[t - T_{mts}]^+ \). With the arrival curves and service curves, the end-to-end maximum delay can be derived based on the method presented in Section 2.3.4. We use the per-flow based method to derive the end-to-end maximum delay. Next, we compare the calculated delays with the delays obtained from simulations. For simplicity, we select 6 nodes from the whole mesh network for analysis. To not lose generality, the sequence number of these selected nodes are (1,1), (1,2), (2,2), (3,2), (3,3), (4,3) (see Figure 5.7). And the distances from these nodes to the sink are 6, 5, ..., 1 hops, respectively.

\[\text{Figure 5.8. Compare calculated delay with simulated delay (low traffic)}\]

In the delay evaluation experiments, the entire simulation runs until every node has sent 200 packets. In Figure 5.8 and 5.9, "Calculated worst-case delay" is the delay computed by network calculus. "Simulated maximum delay" and "Simulated average delay" are the maximum and average transmission delay of 200 packets from a source node to the sink, respectively.

Figure 5.8 shows, in a low input traffic scenario, how the actual packet transmission delay compares with the calculated delay obtained by network calculus. From this figure, we can see that the simulated maximum delays of all nodes are within close to the calculated worst-case delays. For node
Chapter 5. Case Study

Figure 5.9. Compare calculated delay with simulated delay (heavy traffic)

(1,1), (1,2), (2,2), (3,2), (3,3), and (4,3), the simulated maximum delays are 14.3%, 13.5%, 12.7%, 9.6%, 6.3%, and 6.0% smaller than calculated worst-case delays, respectively. For short paths, the differences between the calculated delays and simulated delays are smaller than those for long paths. The reason is that the ‘real’ worst-case delays for long paths are captured in these simulations. Furthermore, in the figure, all the simulated average delays are much smaller than the calculated worst-case delays. It is because all packets do not experience worst-case delay at the same time.

Figure 5.9 shows, in a heavy input traffic scenario, how the actual packet transmission delay compares with the calculated delay. In this figure, all the simulated maximum delays are closet to those of calculated worst-case delays. The values of deviations are 12.8%, 11.5%, 9.2%, 7.5%, 5.6% and 4.1% for node (1,1), (1,2), (2,2), (3,2), (3,3), and (4,3), respectively. Both in low and heavy traffic scenarios, the differences between the calculated delays and simulated delays for short paths are smaller than that for long paths. Comparing the results, another interesting phenomenon is that the deviations between the simulated delays and the calculated delays for the heavy traffic scenario are smaller than those for the low traffic scenario, i.e.
5.2. Experimental Evaluation

the worst-case delay can be caught more easily in the low traffic scenario.

Normally, sensors are tiny devices with limited hardware resources, such as memory. So it is important to estimate the maximum required buffer size (i.e. backlog) before sensor network deployment. In the following experiments, we compare the calculated worst-case backlog with simulated backlog at each node in two kinds of traffic scenarios. For each experiment, we run the simulation for 5000 time units and record the backlog every time unit.

![Graph showing calculated worst-case backlog and simulated maximum backlog](image)

**Figure 5.10.** Calculated backlog and simulated backlog (low traffic)

Figure 5.10 shows the calculated worst-case backlog (left) and simulated average backlog (right) of all nodes in the low input traffic scenario. From this figure, we can see that, both in the simulation and calculation, as the node is closer to the sink, the backlog is bigger. Therefore, during the designing and deployment of sensor networks, the maximum buffer size should be set according to the nodes nearest to the sink.

Figure 5.11 shows the comparison between calculated backlog and simulated backlog of selected nodes in the low input traffic scenario. From this figure, we can see that the simulated maximum backlogs of all nodes are close to the calculated worst-case backlogs. For node (1,1), (1,2), (2,2), (3,2), (3,3), and (4,3), the simulated maximum backlogs are 2.4%, 4.5%, 5.3%, 6.7%, 6.4%, and 7.9% smaller than calculated worst-case backlogs, respectively. For nodes far from the sink, the differences between simulated backlogs and calculated backlogs are smaller than those for nodes near to the sink. The reason is: for the nodes near to the sink (i.e. downstream nodes), the estimation of backlog is impacted by the traffic generated by itself and
the estimated traffic from the upstream nodes. Since the estimated traffic is not as accurate as the actual one, so variation becomes bigger if there are more upstream nodes. Furthermore, in the calculation the estimated traffic from the upstream nodes is its upper bound traffic, while this worst case does not happen in our simulations. If we want to capture the worst cases by simulations, much more runs are needed.

Figure 5.12 shows the calculated worst-case backlog (left) and simulated average backlog (right) of all nodes in the heavy input traffic scenario. From this figure, we can see that, both in simulation and calculation, as the node is closer to the sink, the backlog is bigger. This trend is the same as that in the low traffic scenario.

Figure 5.13 shows the comparison between calculated backlog and simulated backlog of selected nodes in the low input traffic scenario. In this figure, we can see that all the simulated maximum delays are smaller than those of calculated worst-case delays. The values of deviations between them are 2.2%, 4.1%, 3.9%, 5.3%, 6.4%, and 7.5% for node (1,1), (1,2), (2,2), (3,2), (3,3), and (4,3), respectively. The same as that in the low input traffic sce-
5.2. Experimental Evaluation

Figure 5.12. Calculated backlog and simulated backlog of all nodes (heavy traffic)

Figure 5.13. Comparison between calculated backlog and simulated backlog of selected nodes (heavy traffic)

In this scenario, the differences between simulated backlogs and calculated backlogs for nodes far from the sink are smaller than those for nodes near to the sink.
5.3 Summary

In summary, we can get the following conclusions from the experiment on the chain scenario and mesh scenario. Overall, all the simulation results are within the range of the calculation results. And the network calculus based method is accurate in estimating the worst case delay and backlog. For the delay estimation, if the distance (i.e. number of hops) from the source to the sink is smaller (less than 5 hops), the differences between the calculation results and simulation results are also smaller. For the backlog estimation, if the path is longer, the deviations between the calculation results and simulation results are bigger. Moreover, both for the delay and backlog, the differences between the results obtained by network calculus and simulation are bigger for the low input traffic scenario than those for the heavy input traffic scenario. The reason is: for long paths, the chance that simulations could capture the worst-case results is smaller than that for short paths. Similarly, if the input traffic load is heavier, the worst-case results can be caught by simulations more easily.
Chapter 6

Conclusions and Future Work

This chapter concludes the thesis and outlines future directions.

6.1 Conclusions

Due to its characteristics, wireless sensor network design presents several challenges while dealing with various requirements and diverse constraints. To provide insight on the design parameters on system behavior, such as data transmission delay and maximum required buffer, performance analysis techniques are required. In this thesis, we have presented a performance analysis method for WSNs based on network calculus. With this method, some performance metrics can be analytically evaluated rather than by case-by-case simulations.

We define three general traffic flow operators to model any traffic flowing scenarios in sensor networks. The worst-case delay bound and backlog bound of these operators are separately analyzed using network calculus. Based on the results of general operators, a deterministic worst-case performance analysis method is presented. Providing network parameters, our method can provide an effective way for a designer to estimate the worst-case performance and buffer cost of sensor networks. Moreover, we have proposed a method to analyze traffic splitting mechanisms in mesh sensor networks and generalized sensor networks. In addition, we have presented an analytical technique to evaluate the maximum transmission delay and energy consumption of two categories of retransmission schemes: hop-by-hop retransmission and end-to-end retransmission.
In order to validate the tightness of the two bounds obtained by calculating, we have carried out several experiments using Omnet++ based on the case study of a fresh food tracking system. The simulation results and analytical results are compared in the chain and mesh scenarios with various input traffic loads. From the results, we show that the network calculus can be useful and accurate for performance analysis of WSNs.

6.2 Future Work

Our future work may focus on the following aspects:

- Reliable transmission is essential for many applications of sensor networks, such as health care and intrusion detection. However, higher reliability requires more power consumption of sensors, and thus the lifetime of the network would be reduced. It is important to investigate the trade-offs between reducing power consumption and improving transmission reliability for these applications.

- Fault tolerance is a crucial issue in many applications of sensor network. How to integrate fault-tolerance into sensor network calculus would be an interesting research topic.

- The wireless link is unreliable due to several factors such as interference, attenuation and fading. In order to capture the random and dynamic behaviors of WSNs, we are going to apply stochastic network calculus [29, 38] to investigate such issues in the future.
References


