Modeling and evaluation of the end-to-end delay in wireless sensor networks

Francois Despaux

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Modeling and evaluation of the end-to-end delay in wireless sensor networks

THÈSE

présentée et soutenue publiquement le 25 Septembre 2015

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

François Despaux

Composition du jury

Président :	Le président	
Rapporteurs :	Thomas Noël	Pr. à l'Université de Strasbourg
	Congduc Pham	Pr. à l'Université de Pau et des pays de l'Adour
Examinateur :	Thierry Val	Pr. à l'Université de Toulouse
Encadrants :	Ye-Qiong Song	Pr. à l'Université de Lorraine
	Abdelkader Lahmadi	MCF à l'Université de Lorraine

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Acknowledgments

Il me sera très difficile de remercier tout le monde car c'est grâce à l'aide de nombreuses personnes que j'ai pu mener cette thèse à son terme.
Je tiens tout d'abord à remercier mes encadrants Ye-Qiong Song et Abdelkader Lahmadi pour leur disponibilité, leur capacité et leur grande qualité professionnelle dont ils ont fait preuve lors de ces quatre ans de thèse. Leur charisme et leur côté humain ont permis, dans un premier temps, l'établissement d'un lien de confiance et respect mutuel forgeant par la suite une excellente relation, pas uniquement sur le plan professionnel, mais aussi humaine. Je les remercie de m'avoir guidé, encouragé, conseillé tout en me laissant une grande liberté et en me faisant l'honneur de me déléguer plusieurs responsabilités dont j'espère avoir été à la hauteur. Un grand merci à eux.
Je remercie également les membres de l'équipe Madynes, ceux qui sont encore là ainsi que ceux qui sont déjà partis pour différents raisons.
Je tiens à remercier les assistantes, Delphine Hubert, Céline Simon et Isabelle Herlich pour leur aide sur le plan administratif.
Je remercie également les Professeur Thomas Noel, Professeur Congduc Pham et Professeur Thierry Val d'avoir expertisé mon travail.
Mes derniers remerciements vont à Julie qui m'a soutenu et surtout supporté ces derniers temps.

Je dédie cette thèse à ma mère, mon père et mes frères.

Contents

List of Figures

Chapter 1

Introduction

Contents

1.1 Introduction

Wireless Sensor Networks (WSNs) are a class of wireless networks intended for monitoring environmental phenomena in a given physical space. A WSN is composed of small devices with multiple on-board sensors which can communicate with each other through wireless links. Typical sensor examples include temperature, light, sound and humidity. Sensors read or sense the environment and transmit the gathered information over a wireless channel to a base station or directly to actuators in order to take further decisions according to the received data. When actuators are present the network is referred as a Wireless Sensor and Actuator Network (WSANs) which is an important extension of WSNs. A sensor network is composed of hundreds and even thousands sensor nodes. Each of these nodes is composed, typically, of several parts: a radio transceiver with an internal antenna, a microcontroller, on-board sensors, a memory and power supply, normally a battery. A sensor node is resource constrained, specially in terms of the power supply since the overall battery capacity is limited. Components such as memory, microcontroller, radio transceiver should be configured in such a way that the energy consumption is optimized. Therefore, the computation capacity, memory and communication on each sensor is limited. Besides, MAC protocols are designed taking into account the fact that the energy consumption must be optimised. In this way, mechanisms such as duty-cycle are implemented in order to save energy in nodes and to extend the network lifetime.
The development of WSN has taken traditional network topologies. They are the peer to peer, star and tree topologies. Peer to peer networks allow each node to communicate directly with another node without needing to go through a centralized communications coordinator or hub. Then, each peer device works as both a sender and receiver to the other nodes on the network. In the context of a star topology, nodes or devices connect with a centralized node or coordinator in such a way that the entire communication must be routed through the centralized node. Nodes

or devices cannot communicate directly to each other. Each node or device is then a client and the coordinator acts as a server. Tree topology consists of a central node or root as the main communication router. Next below levels in the hierarchy follow a parent-children communication scheme.
Several applications have been envisioned for WSNs because of the inherent advantages it provides, each of them with different requirements in terms of the performance parameters. Typical applications of WSNs can be:

Military Applications WSNs can be critical part of intelligence, surveillance, reconnaissance and targeting systems. Some examples where WSNs can be deployed are nuclear, biological and chemical attack detection, monitor of enemy forces where troop and vehicles can be tracked in detail. A survey of military applications in WSNs can be found in [Durisic et al., 2012]. As an example, a set of spread sensor in a battlefield may help in retrieving information regarding the enemie's positions and send the aggregated information to soldiers in order to strategically find the positions themselves to minimize the risks. As long as tracking is concerned, real-time constraints appear, in addition to energy constraints.

Environmental Applications Several environmental applications have been proposed using WSNs [Ye et al., 2009]. Some examples are habitat monitoring, animal tracking, precision farming, forest fire detection among others. Requirements may vary depending on the type of the application. A forest fire detection system, for instance, may have strict requirements in terms of the delay since the information must be transmitted immediately when a fire is detected in order to take actions accordingly. On the other hand, an habitat monitoring system may require a good performance in terms of energy consumption since this kind of applications must run for a long period of time.

Health-care Applications WSNs can also be used in health-care applications. Sensors placed on patient's body may help monitoring the vital signs and also track their location and alert if any unusual issue arises. As an example, a potential application suitable for WSN is the one related to glucose-level monitoring for patients with diabete disease where patients require constant monitoring of blood sugar levels to lead healthy. Sensor's information regarding the blood-sugar levels and trends can be sent to a doctor and, if necessary, actions to normalize the blood-sugar level can be taken in timely manner before reaching critical levels. Through these examples, we can see that a WSN should, not only provide energy efficiency, but also end to end delay guarantee, specially when real-time applications are concerned.

In this thesis, we addresse the problem of estimating the end to end delay for multi-hop scenarios in a WSN. Having a way for estimating the end to end delay is important since there are many applications in the domain of WSNs which are delay-sensitive. In the next subsection we detail some of them.

1.1.1 Delay-sensitive Applications in WSNs

In this thesis, we focus our attention in those WSN's applications that require bounded end to end delays with a certain probability. The performance-critical applications that require bounded delay are referred to as real-time applications. Health care applications described above, are an example of this kind of applications where sensors send information to a health-care provider

regarding the patient's health status, alerting whenever something goes wrong. Another important domain of real-time applications concern the industrial process monitoring and control applications. Specifically, in such applications, process data such as pressure, humidity, temperature, viscosity, density and vibration intensity measurements can be collected through sensors and transferred to a control system to take actions accordingly. Process monitoring and control is a combination of architectures, mechanisms and algorithms used in the industrial factory for monitoring and control the activities of a specific process to achieve a given goal. As an example, cooling down a reactor by adjusting the flow rate through the cooling jacket is a process that has a specific goal: to keep a constant predefined temperature over the time. Most of the process control applications are mission critical and have stringent requirements. Failure of a control loop may cause unscheduled plant shutdown or even severe accidents in process-controlled plants. Normally, these applications have requirements in terms of quality of service, specially, to send the correct data at the right time (reliability) and the real-time guarantees (latency). As an example of the real-time requirements, the real time temperature information in an industrial process control is used to decide the adjustment of a valve to manipulate the flow rate through the cooling jacket. In this kind of applications, excessive delay may cause overshoot or disturbance to the process.
Another example of applications having requirements in terms of delay concern the early warning system as the one deployed in Japan for detecting earthquakes and tsunamis [EWS, 2007]. These applications rely on a set of sensors deployed in some area whose objective is to early detect potential earthquakes. Upon detecting some unusual behavior, they send the information as fast as possible to warning centers which will take decisions before the moment the earthquake hits, for instance, by delivering alerts to the population including emergency management personnel. It is clear that, for this kind of applications, it is important to have a way for estimating the end to end delay in order to be able to efficiently meet the application requirements. However, and due to the stochastic nature of WSNs and the underlying MAC protocols, estimating the end to end delay in a precise manner is not possible. Instead, these applications require that the end to end delay should not exceed a given threshold with a certain probability, requiring thus a probabilistic quality of service (QoS) guarantee.

Our objective in this thesis is to be able to find this probabilistic estimation of the end to end delay. Concretely, we look for the estimation of the probability distribution of the delay in a WSN from a given source node to a destination node in a multi-hop scenario.
There are many works in the literature addressing this problem, an overview of them are summarized below.

1.1.2 Common Approaches for Evaluating End to End delay

Many work has been done during the years for estimating performance parameters in WSNs, specially, the delay. Basically, three approaches are proposed in order to estimate this performance parameter: measurement, simulation and analytical models, each of them with its advantages and disadvantages. The measurement approach consists in setting up a scenario in a real WSN testbed and to measure the end to end delay from a given node (source) to the destination one. Having ideal time synchronisation component (high precision with negligible overhead), the end to end delay measurement can be simply achieved by taking two timestamps on both source and destination node. However, in a large scale network, global time synchronisation is very costly in both the bandwidth and program memory consumption. Besides, the synchronisation service usually suffers a lot from the unreliable wireless links as well as the environmental interferences.

Due to the difficulty to achieve global synchronisation in large-scale sensor networks, measuring the end to end delay is extremely challenging. Some open platforms, such as IOT-LAB [Fambon et al., 2014], provide time synchronisation through parallel networks (e.g. Ethernet).
A second approach used to solve this problem is by means of simulation tools. The nature of WSNs makes difficult to analytically model a WSN and predict the actual performance of protocols and network operation. Moreover, deploying and operating a testbed to study the behavior of protocols and network performance supposes a great effort. Consequently, simulation is essential to study WSN, being the common way to test new applications and protocols. The advantage of simulation over measurements is the simplicity for setting scenarios with a large number of nodes (hundred or thousands). Besides, simulations can overcome the problem of clock synchronisation found in measurement approach. However, it requires a suitable model based on several assumptions (hardware and physical layer) and an appropriate framework to ease implementation.
Finally, many authors propose analytical models for modelling the behavior of a WSN and mainly its MAC associated protocols. Due to the stochastic nature of MAC protocols, the solution is mainly based on Markov chain models. Based on these models they are able to estimate the most important performance parameters. However, capturing the real behavior of a WSN is not straightforward. Most of the proposed models simplify the reality by considering ideal channel conditions, a simplified queue model on each node, Markovian traffic, etc. leading then to models that are not enough accurate for estimating the delay. Most of these models found in literature were conceived for the IEEE 802.15.4 MAC protocol since it is the standard one [Misic and Misic, 2008], [Park et al., 2009]; only a few work can be found for other MAC protocols such as S-MAC and X-MAC [Yang and Heinzelman, 2012]. However, these models are limited to a star topology. Hence, delay is estimated in a one-hop fashion and its extension to multi-hop cases for estimating the end to end delay is not possible due to the impossibility to characterize the input traffic arriving to intermediate nodes. Normally, these models suppose Markovian traffic for modelling the queue behavior. Even though this can be assumed for the first (source) node, it is not possible to apply the same model for intermediate nodes since the input traffic distribution is unknown. [Wang et al., 2012] is one of the rare work dealing with multi-hop network and authors are able to estimate the end to end delay in a multi-hop scenario considering the TinyOS default CSMA/CA MAC protocol (similar to IEEE802.15.4). However, all nodes in the network are modeled by a Geom/PH/1/M queue. In other words, the input traffic distribution to all nodes (even the intermediate nodes) is supposed to be Geometric, an issue which is not verified in general. Another important issue regarding the existing analytical models is the fact that they only focus on modelling the MAC protocol logic, without considering the network implementing this protocol, i.e., without considering the impact of having an operating system (OS) running on each node. We have verified in [Despaux et al., 2013], that the OS introduce non-negligible extra-delays that will affect the one hop delay and therefore, the end to end delay.
Our goal in this thesis is to propose an approach for infering a model for a WSN in order to be able to estimate the end to end delay from a source node to the destination in a multi-hop scenario. We propose then a way for extracting a Markov model overcoming the problems mentioned before. Next, we details the contribution of our work.

1.1.3 Contributions

In this thesis, we propose a novel approach for modelling the network behavior combining measurement and analytic approaches to overcome the main problems described above. Our approach

consists in infering a Markov chain model by analysing the MAC protocol execution traces. The main contributions of this work can be enumerated as follows:

- We propose a novel methodology to infer a Markov chain by analysing the execution traces of a given MAC protocol. Since this Markov chain is obtained by analysing the execution traces, factors like the impact of the underlying operating system are taken into account, an issue not considered in existing analytical models.
- Contrarily to the existing approaches where the analysis is done for a particular MAC protocol (e.g., the IEEE 802.15.4 MAC protocol), our methodology is applicable to any underlying MAC protocols.
- Contrarily to the existing Markov chain models, where the assumptions in terms of the distribution of the packet arrival for modelling the node behavior must be considered (Markovian traffic), our approach does not make any assumption regarding this distribution. Hence, this approach allows to model the behavior of intermediate nodes without strong assumptions on the packet arrival distribution.
- Based on this Markov chain, we are able to estimate the one hop delay distribution for each of the nodes in the network (even for intermediate nodes) from the moment a packet arrives to the node until the corresponding acknowledgement is received. Then, we extend the analysis for estimating the end to end delay in a multi-hop transmission scenario by means of frequency domain analysis.
- A generalisation of this approach based on non-linear regression techniques that allows us to estimate the end to end delay for unknown arrival rates is also presented.

Our approach was validated for three MAC protocols: ContikiMAC [Despaux et al., 2014a], IEEE 802.15.4 MAC [Despaux et al., 2014b] and X-MAC [Despaux et al., 2015] and based on the results, we can conclude that our approach is suitable for estimating the end to end delay in multi-hop WSNs.

1.2 Manuscript Organization

This manuscript is organized in 7 chapters. We start by giving an introduction (this chapter) to WSNs and its applications and putting this thesis in context. Second and third chapters give the background necessary to understand the problem we are focused on, and present the related work concerning the estimation of the one hop delay in WSNs, emphasising on the typical analytical approaches found in the literature for modelling the stochastic behavior of a WSN. Testbed experiments regarding the analysis of one-hop delay in WSNs are presented in chapter 4. Chapter 5 presents our approach for modelling and estimating the end to end delay distribution in a WSN, an approach that overcomes the main problems of previous work found in the literature. Chapter 6 presents a generalisation of the proposed approach based on non-linear regression techniques in order to be able to estimate the end to end delay independently of a specific packet arrival rate. Finally, conclusions and future work are presented in chapter 7.

Chapter 2

Background

Contents

2.1 Introduction

The objective of this chapter is to give the context and background necessary to understand the problem we are focused on. We start first by giving a definition of the end to end delay in a WSN since it is the most important issue adressed in this thesis. Next, we explain the mechanism of duty cycle, a mechanism omnipresent in most commonly used MAC protocols whose purpose is to save energy in nodes and thus in the whole WSN. As we will see next, this mechanism for saving energy in nodes has its cost, specially in terms of delay, since nodes periodically enter in sleep mode and are not always available for transmitting/receiving packets. Hence, it will impact on the whole end to end delay. Since we are focused in studying the delay in WSNs, the most important factors affecting both one hop and multi-hop delays are also presented. We give also an introduction to the IEEE 802.15.4 protocol since it is the standard MAC protocol for WSNs together with the most commonly used MAC protocols in WSNs.
Finally, an overview to the routing protocol RPL is also presented since it is considered in our experiments in this thesis.

2.2 End to end delay in WSNs

The end to end (e2e) delay in a WSN is defined as the delay from the moment a source node has a data packet ready to send until the moment this packet reaches the destination node. Due to the nature of a WSN, giving delay guarantees is not always straightforward. Normally, delay guarantee is achieved through providing deadlines to each individual message at the network and the application level. Recent work has analyzed the latency performance of WSNs in terms of its first-order statistics (mean and variance) [Abdelzaher et al., 2004], [Bisnik and Abouzeid, 2006], [Gupta and Shroff, 2009]. However, complex and cross-layer interactions in multihop WSNs require a complete stochastic characterization of the delay. Several works focused in providing probabilistic *bounds* on delay. The concept of network calculus [Cruz, 1991] has been extended to derive probabilistic bounds in terms of delay through worst-case analysis [Burchard et al., 2006], [Fidler, 2006]. However, worst-case analysis cannot capture the stochastic behavior of the end to end delay. Some other works providing stochastic models for unreliable networks known as real-time queueing theory, were presented in [Lehoczky, 1997]. However, they consider heavy traffic condition which is not applicable to WSNs. Authors in [Serna Oliver and Fohler, 2009] give a novel concept of delay guarantees defining it as the acceptable interval for sequence of packets to be delivered with a defined level of confidence (probability) rather than a strict end to end deadlines for individual packets. Then, two types of real-time guarantee can be distinguished: hard real-time (HRT) and soft real-time (SRT). In HRT, a deterministic end to end delay bound must be assured. On the other hand, in SRT a probabilistic guarantee is required and some deadline miss is tolerable. In this thesis, we are interested in finding a probabilistic estimation of the e2e delay instead of a strict upper-bound on delay.

In general the delay is influenced by a set of parameters such as the distance and hop-count toward the destination node, the nodes density, data rate, node's resources, MAC, and routing communication protocols. All these parameters can provide high and unpredictable e2e delay. From the stack point of view, MAC layer affects the one hop delay while the network layer the multi-hop e2e delay. At the MAC layer, the one-hop delay can be expressed as:

$$\begin{aligned} OneHop_{delay} = Processing_{delay} + Queuing_{delay} + \\ Channel\_Access_{delay} + Transmission_{delay} + \\ Propagation_{delay} + Reception_{delay} \end{aligned} \tag{2.1}$$

In other words, the interval from the moment a packet is generated in the node (or arrived from some neighbor node) until de corresponding acknowledgement is received from the next hop node receiving the packet. The one-hop delay components are shown in Figure 2.1. Here, *Queuing* and *Channel_ Access* delay are the most important components in the whole one hop delay since the other ones are hardware dependent. This is the main reason why current researches focus on improving queueing strategies and channel access techniques to ensure good performance in terms of latency. On the other hand, from the routing layer stand point, the end to end delay can be computed as:

$$EndtoEnd_{delay} = \sum_{i=1}^{k} (OneHop_{delay}(i) + \mathcal{C}) \tag{2.2}$$

where $i \in \{1, ..., k\}$ are the nodes within the selected path from source to destination and $\mathcal{C}$ is a bounded value representing the delay, for each hop, of the time from the moment a neighbor packet arrives until it is added to the MAC queue for retranmission, which means to cross in a bottom-up fashion each of the network layers until reaching the application layer.

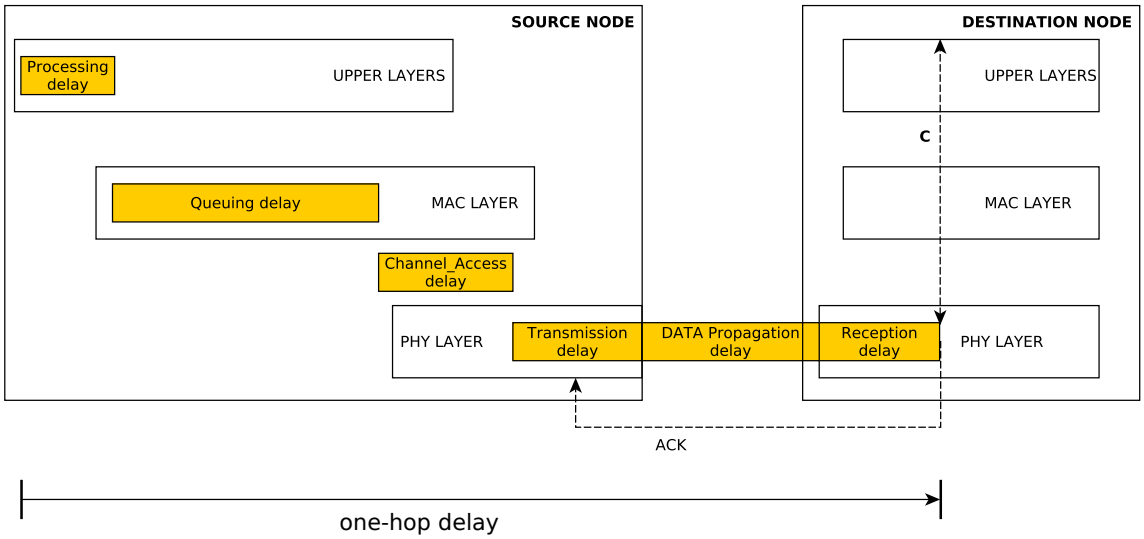


Figure 2.1: One hop delay components.

2.3 Major Factors Affecting Delays in WSNs

Regarding the MAC layer, a set of factors affecting the one-hop delay can be identified.

Duty Cycle At the MAC layer, there are many factors affecting both energy and delay. One of the most important factors affecting the delay is the *duty cycle* mechanism, implemented in most commonly used MAC protocols. As sensor nodes are generally battery-powered devices, the critical aspects to face concern how to reduce the energy consumption of nodes, so that the network lifetime can be extended to reasonable times. The basic idea of the duty cycle mechanism is to reduce the time a node is idle or spends overhearing an unnecessary activity by putting the node in the sleep state. The ideal scenario of low duty cycle is when a node is sleeping most of the time and wakes up only to transmit or receive packets. This concept is represented as a periodic wake-up scheme; a node wakes up, it listens to the channel for any activity before transmitting or receiving packets. If no packet is to be transmitted or received, the node returns back to sleep mode. A whole cycle consisting of a sleep period and a listening period is called a sleep/wake-up period and is depicted in Figure 2.2. Duty cycle is measured as the ratio of the listening period length to the wake-up period length which gives an indicator of how long a node spends in the listening period.

Buffer Capacity Each node has a limited queue for keeping those packets arriving to the node which cannot be immediately dispatched. It is important to note that, for a packet arrival rate beyond the buffer capacity, the number of packet loss due to a full queue will increase impacting then the delay for each packet.

Overhearing This happens when a node hears a preamble or data packet for which it is not the intended destination.

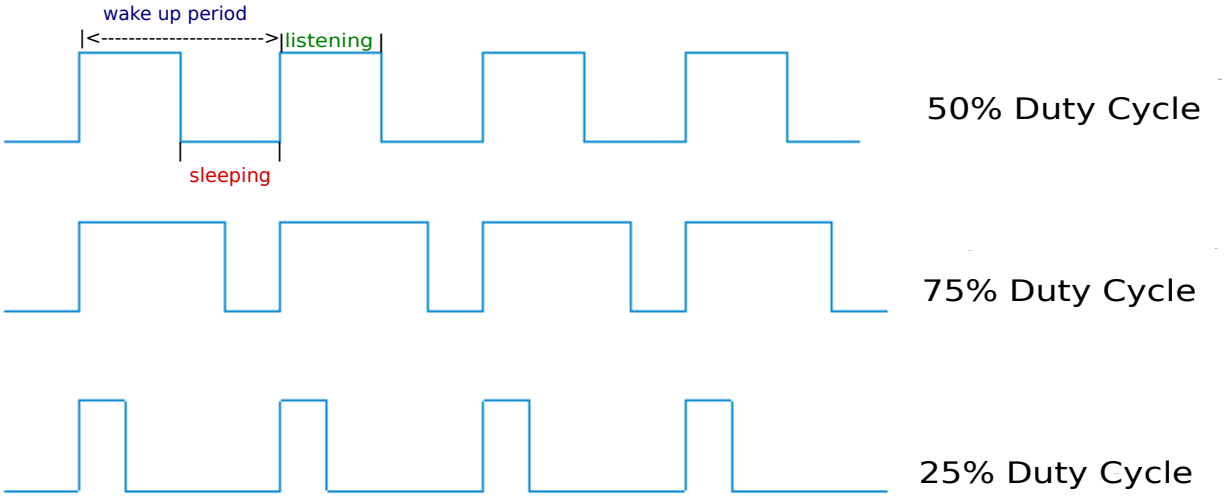


Figure 2.2: A periodic wake-up scheme: 25, 50, 75%.

Over-emitting As we will see further in asynchronous MAC protocols, sending data for a destination node which is not ready to receive the packet will incurs in an extra local delay local delay since the packet should be retransmitted after a backoff period.

Collisions Collision may happens when two nodes transmit packets at the same time and interfere with each other's. At the MAC layer, protocols normally implement a mechanism the backoff which is used each time a collision happens and whose objective is to retransmit the packet after a random period. In other words, each time a collision happens the node enters in a random backoff period in order to try retransmitting the packets afterwards. This mechanism will affect the service time delay.

Regarding the routing layer, the fact of having paths that are dynamically changing all time, will impact in the whole end to end delay from source to destination nodes. In other words, the impact of the MAC layer (and thus, the MAC protocols) with respect to delay can be seen as locally (in one node) while the scope of the Routing layer (and thus routing protocols) cover the whole wireless sensor network.

2.4 IEEE 802.15.4 Standard Protocol

The IEEE 802.15.4 protocol [802, 2007] specifies the Medium Access Control (MAC) sub-layer and physical layer for Low-Rate Wireless Private Area Networks (LR-WPAN) and is intended to be suitable for wireless sensor networks since they can be built up from LR-WPANs. Important features include real-time suitability by reservation of guaranteed time slots and collision avoidance through CSMA/CA. Devices also include power management functions such as link quality and energy detection. We are going to see this protocol in detail since it is one of the most important MAC protocols for WSNs. Moreover, most of existing theoretical models for

modelling the behavior of WSNs are based on this standard protocol. Hence, it is relevant to make a clear description on how this protocol works.

2.4.1 IEEE 802.15.4 Medium Access Control

General Description

The idea of the MAC sub-layer in the IEEE 802.15.4 protocol is to provide an interface between the physical layer and the higher layer protocols of LR-WPANs. Like the IEEE 802.11 protocol, the standard makes use of CSMA/CA (Carrier sense multiple access with collision avoidance), a multiple access method in which carrier sensing is used, as the channel access protocol and it also brings support for contention-free and contention-based periods. Table 2.1 summarizes the most important constants and attributes of the IEEE 802.15.4 MAC protocol. As shown in Figure 2.3, the protocol provides two main operational modes: *beacon-enabled mode* where beacons are periodically generated by the coordinator to synchronize attached devices and *non beacon-enabled* mode.

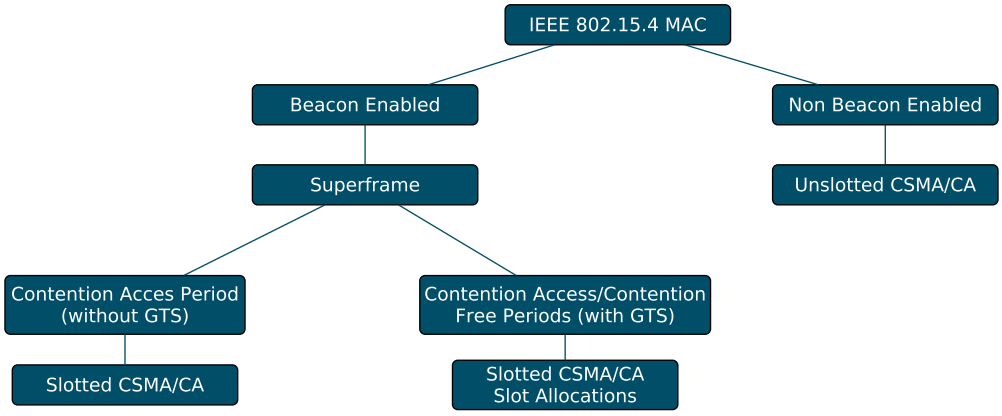


Figure 2.3: Operational Modes of the IEEE 802.15.4 MAC protocol

Beacon-enabled mode This is the synchronous version of the IEEE 802.15.4 protocol. In this mode a *superframe* structure is proposed in order to manage the communication between devices. The superframe format is defined by the PAN coordinator and is sent to the other devices periodically. The superframe structure is shown in Figure 2.4. It is bounded by two consecutive beacon frames and includes one *Contention Access Period* (CAP) and optionally a *Contention Free Period* (CFP). If no CFP is specified, devices that wish to communicate within the CAP period must compete with the others by means of a slotted CSMA/CA mechanism. All transmissions must be finished before the end of the superframe, i.e. before the beginning of the inactive period. In case that CFP is specified (due to some QoS requirements), *Guaranteed Time Slots* (GTSs) must be allocated by the PAN coordinator to some nodes in order to meet low-latency or specific data bandwidth requirements. The CFP is a part of the superframe and starts at a slot boundary immediately following the CAP. CFP is also shown in Figure 2.4. All values shown in 2.4 were taken by considering a bit rate of $250kbps$. The PAN coordinator may allocate up to 7 GTSs. With this superframe configuration, all contention-based communication must be finished before the start of the CFP. Devices transmitting within a GTS must ensure that its transmission will finish before the beginning of the next CFP (or the end of the CFP).

Constant Name	Description	Default Value
aMaxBE	The maximum value of the backoff exponent in the CSMA/CA algorithm	5
aUnitBackoffPeriods	The number of symbols forming the basic time period used by the CSMA/CA algorithm	20
aBaseSlotDuration	The number of symbols forming a superframe slot	3 × *aUnitBackoffPeriod* = 60
macBattLifeExt	Indication of whether battery life extension, by reduction of coordinator receiver operation time during the CAP, is enabled. A value of TRUE indicates that it is enabled	FALSE
macBeaconOrder	Specifies how often the coordinator transmit a beacon. The macBeaconOrder, BO, has a value of 0 <= BO <= 14. If BO = 15, the coordinator will not transmit a beacon.	15
macMaxCSMABackoffs	The maximum number of backoffs the CSMA/CA algorithm will attempt before declaring a channel access failure.	4
macMinBE	The minimum value of the backoff exponent in the CSMA/CA algorithm. If this value is set to 0, collision avoidance is disabled during the first iteration of the algorithm. Also, for the slotted version of the CSMA/CA algorithm with the battery life extension enabled, the minimum value of the backoff exponent will be the lesser of 2 and the value of *macMinBE*.	3
macSuperframeOrder	This specifies the length of the active portion of the superframe, including the beacon frame. The macSuperframeOrder, SO, has a value of 0 <= SO <= 14. If SO = 15, the superframe will not be active following the beacon.	15

Table 2.1: IEEE 802.15.4 MAC constants and attributes.

In both configuration the superframe structure can have an *inactive period* during which devices may enter in a low power mode (for example, sleep mode) to save energy. The structure of the superframe is defined by two parameters:

- *macBeaconOrder*(BO): determines the interval during which the coordinator must transmit beacon frames. Relationship between *macBeaconOrder* and the *Beacon Interval* (BI) is:

$$BI = aBaseSuperframeDuration * 2^{BO} \ , \quad 0 <= BO <= 14 \tag{2.3}$$

- *macSuperframeOrder*(SO): which describes the length of the active portion of the superframe. The relationship between *macSuperframeOrder* and the *Superframe Duration* (SD) is:

$$SD = aBaseSuperframeDuration * 2^{SO} \ , \quad 0 <= SO <= 14 \tag{2.4}$$

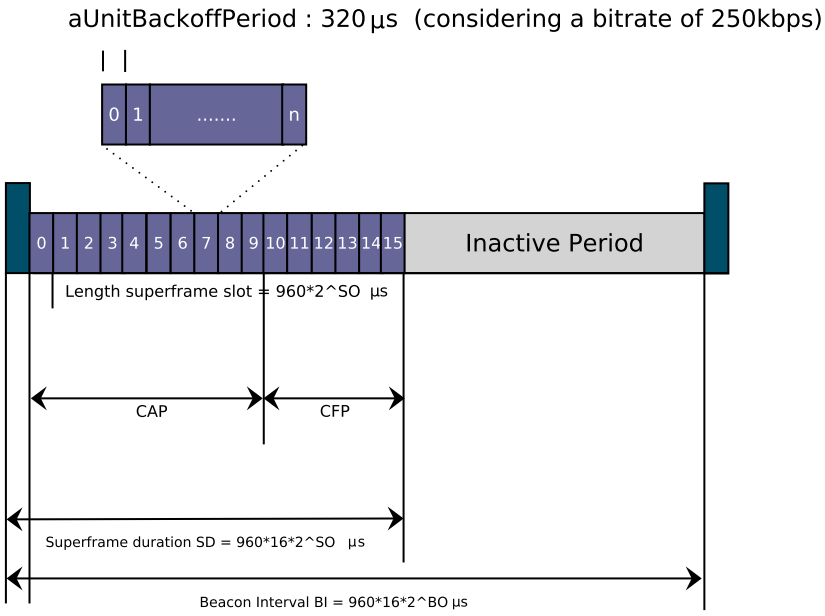


Figure 2.4: Superframe Structure

where

$$aBaseSuperframeDuration = aBaseSlotDuration * aNumSuperframeSlots \tag{2.5}$$

where *aNumSuperframeSlots=16* represents the number of equally spaced slots of the active portion of the superframe. *aBaseSlotDuration* represents the number of symbols (60 syms) forming a superframe slot when the superframe order is equal to zero (*1 syms = 4 bits*). One slot contains 3 backoffs since the constant *aUnitBackoffPeriod* is equals to 20 symbols. The active portion of the superframe structure is composed by:

- *Beacon*: transmitted without the use of CSMA at the start of slot 0. It contains the information on the addressing fields, the superframe specification, the GTS fields, etc.
- *CAP*: starts immediately after the beacon frame and ends before the beginning of the CFP (if it exists). Otherwise it finishes at the end of the active part. The minimum length of the CAP is fixed by the *aMinCAPLength = 440 symbols*. All the transmissions within the CAP are made by using slotted CSMA/CA to access the channel. Acknowledgement frames of the next superframe.
- *CFP*: starts immediately after the end of the CAP and must be completed before the start of the next beacon frame. Transmissions in CFP are contention-free and do not use CSMA/CA mechanism to access to the channel.

Non-beacon enabled mode In this mode, neither beacon nor superframes are considered. Medium access control is provided by an unslotted CSMA/CA. Contrarily to the beacon-enabled mode, this operational mode is asynchronous.

The CSMA/CA Mechanism

The IEEE 802.15.4 defines two versions of the CSMA/CA mechanism: The *slotted CSMA/CA* used in the beacon-enabled mode and the *unslotted CSMA/CA* used in the non beacon-enabled mode. In both cases the CSMA/CA mechanism is based on backoff periods where one backoff period is equal to *aUnitBackoffPeriod=20 symbols*.
Three variables are considered in the CSMA/CA mechanism:

- *NB*: represents the number of times the CSMA/CA algorithm will enter in backoff while attempting the access to the current channel. It is initialized to zero before each new transmission attempt, and increases following each backoff.
- *CW*: represents the number of times the CSMA/CA will check the channel availability before starting transmission. The value by default is 2 and is reset to 2 each time the channel is busy. It decreases after each channel clear.
- *BE*: represents the backoff exponent. Each time the channel is found busy BE is incremented by 1 until it reach the maximum possible value *aMaxBE* which is a constant defined in the standard it has a default value equal to 5. This parameter is used for setting a random waiting delay for collision avoidance as we will see further on.

Figure 2.5 shows a scheme for both slotted and unslotted CSMA/CA mechanism.

The Slotted CSMA/CA Mechanism

- *1 - Initialization*: In this step, NB, CW and BE are initialized. BE value is determined based on the *macBattLifExt* parameter. If its value is equal to *true* then BE is initialized as $BE = min(2, macMinBE)$, otherwise, it is initialized as $BE = macMinBE$ where *macMinBE* specifies the minimum of backoff exponent which is set to 3 by default. After the initialization, the algorithm locates the boundary of the next backoff period.
- *2 - Random waiting delay for collision avoidance*: In this point the algorithm waits a random backoff in order to avoid collisions. The random backoff period is taken from the range of $[0, 2^{BE} - 1] \times$ *aUnitBackoffPeriod*.
- *3 - Clear Channel Assessment*: In this step the algorithm checks the availability of the channel. If the channel is found idle the algorithm goes to the *idle channel* step, otherwise it moves to the *Busy channel* step. The CCA must be started at the boundary of a backoff period after the expiration of the waiting delay timer.
- *4 - Busy channel*: In this step, the CCA found the channel busy. Then, BE and NB parameters are incremented by 1 and CW is reset to the initial value. The BE parameter must not exceed the value of *aMaxBE* parameter whose value by default is 5. If NB exceeds the parameter *macMaxCSMABackoffs* then the algorithm finishes by throwing a channel access failure status. Otherwise, if NB is lower or equal to *macMaxCSMABackoffs*, the algorithm jumps to step 2 (*Random waiting delay for collision avoidance* step).
- *5 - Idle channel*: In this step, the channel was found idle. Then, CW parameter is decremented by 1. In case CW reaches zero, the MAC protocol may start successfully its transmission. Otherwise, the algorithm returns to step 3. The transmission is started if and only if the remaining number of backoff periods in the current superframe is sufficient to handle both the data packet and the subsequent acknowledgement transmissions. Otherwise, transmission is deferred until the next superframe.

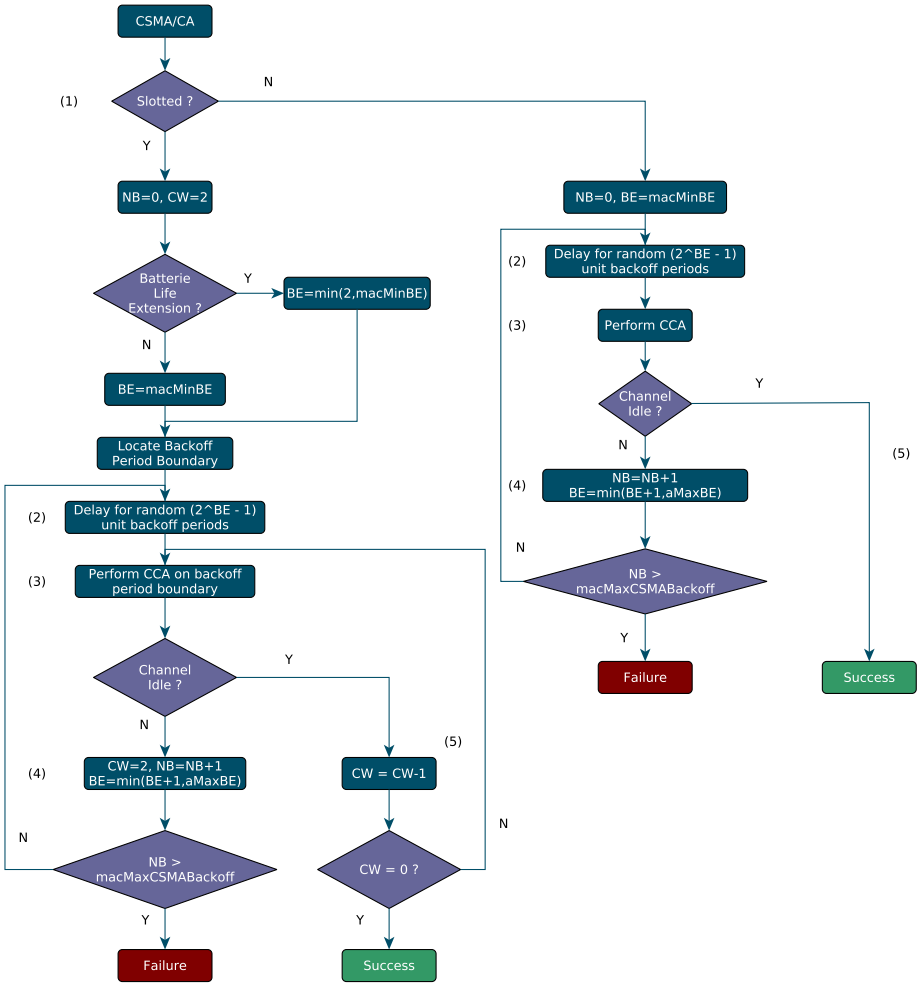


Figure 2.5: IEEE 802.15.4: Slotted and Unslotted Version

The Unslotted CSMA/CA Mechanism This mechanism is similar to the previous one with some few exceptions.

- *1 - Initialization*: In this case, parameter CW is not considered. Once the channel is found idle the MAC protocol starts immediately transmitting the data. Besides, this version does not supports *macBattLifeExt* mode so BE is always initialized to *macMinBE* value.
- *Step 2 and 3*: are basically the same as the previous version except that CCA starts immediately after the random backoff expiration.
- *Step 4*: The same as the previous approach but without considering the CW parameter.
- *Step 5*: The MAC sub-layer starts immediately transmitting its current packet just after a channel is assessed to be idle by the CCA procedure.

2.4.2 Network Devices

The IEEE 802.15.4 supports two different types of devices:

- **Full Function Device (FFD)** which can serve as the coordinator of a personal area network. This device identifies its own network to which other devices may be associated. It implements a general model of communication which allows it to talk to any other device. It's a PAN coordinator when it is in charge of the whole network.
- **Reduced Function Device (RFD)** meant to be extremely simple devices with very modest resource and communication requirements. Due to this they can only communicate with FFDs and can never act as coordinators.

A LR-WPAN must include at least one FFD acting as a PAN coordinator that provides global synchronization services to the network and manages potential FFDs and RFDs.

2.4.3 Network Topologies

Basically the standard proposes two types of network topologies: *The star topology* and a *peer-to-peer topology*. Both topologies are shown in Figure 2.6

- The Star Topology: In this topology, a unique node operates as a PAN coordinator. The communication paradigm is centralized in this case, i.e., each device joining the network and willing to communicate with other devices must send its data to the PAN coordinator who will dispatch them to the corresponding devices. PAN coordinator may be main powered while other devices are more likely to be battery powered.
- Peer-to-Peer Topology: It also includes a PAN coordinator. The communication paradigm here defers from the Star Topology since the communication in this case is descentralized and each device can directly communicate with any other. This approach enables enhaced networking flexibility but it introduces an additional complexity for providing an end-to-end connectivity between all devices in the network.

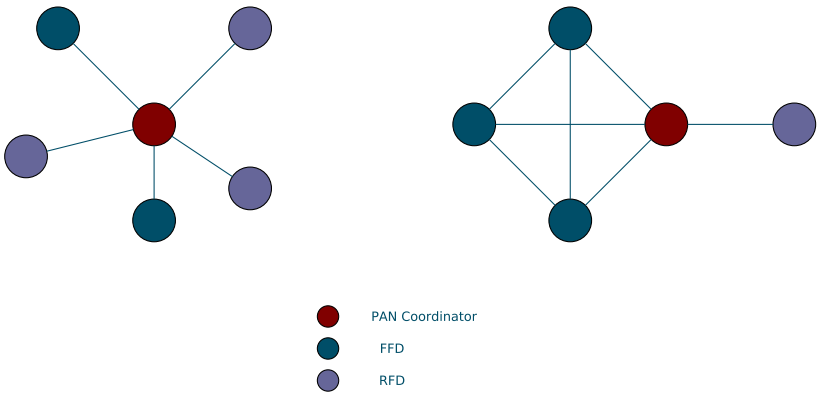


Figure 2.6: Star and Peer-to-Peer Topologies.

We can also consider a third topology named *Cluster-tree* as shown in Figure 2.7. It is a special case of the peer to peer networking in which most devices are FFDs. In this topology, only one coordinator is nominated as the PAN coordinator which identifies the entire network. Any FFD act as a coordinator providing syncrhonization services to the other devices or coordinators. Finally, an RFD connects to a cluster-tree as a leave node at the end of a branch and associates itself with only one FFD.

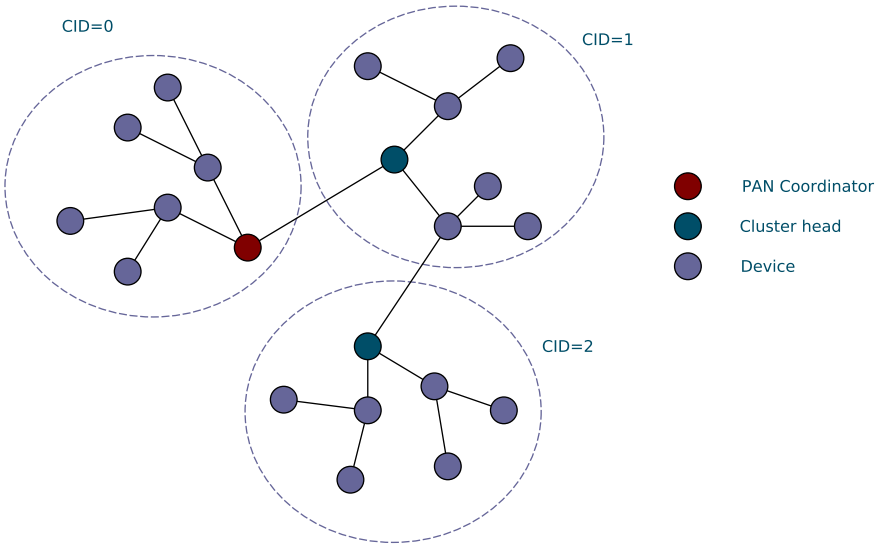


Figure 2.7: Cluster Tree Topology

2.5 Duty Cycle MAC Protocols

Many WSN MAC protocols have been studied in the literature. A list of surveys on MAC protocols can be found in [Demirkol et al., 2006], [Yahya and Ben-Othman, 2009], [Kredo and Mohapatra, 2007] and [Langendoen, 2007]. Kuntz in [Kuntz, 2010] classifies MAC protocols into synchronised protocols, asynchronous (preamble sampling) and hybrid protocols. In synchronous protocols, a node is aware of its neighbors' sleep/active schedules since all nodes in a group or cluster should have the same wake-up phase in order to exchange messages. On the other hand, asynchronous protocols have the advantage over synchronous that there are no requirements in terms of clock synchronisation between nodes.
In this section we detail the main synchronous and asynchronous protocols commonly used in WSN's applications.

2.5.1 Synchronous Protocols

The synchronous scheme is one of the variation of low duty cycle protocols where all nodes in a group or cluster have the same wake-up phase. Normally, each node sends frequent beacon frames to inform its neghbours about its wake-up cycle schedule in order to synchronise with them. In this way, each node schedules its transmission and reception times from the information obtained from the beacons. A variation of this approach is a centralised approach consisting in assigning a node as the head of the group. This node will transmit beacons and will keep the synchronisation between all nodes belonging to the cluster. In these two variations, a node broadcasts its beacon frames once it enters the active period in order to share its current schedule and status information with its neighbouring nodes. Thus, two periods can be identified: T_{sleep} and T_{active} repeated at T_{wakeup_period} interval. This schema was shown in Figure 2.2. As seen, the IEEE 802.15.4 protocol includes this wake-up scheme by means of a superframe structure. Nevertheless, this duty cycle scheme (wake-up scheme) is static in the sense that it does not vary according to the traffic rate. In other words, even in low traffic scenarios, the wake-up scheme does not change. Hence, the energy consumption would be the same whatever the traffic rate is.

Sensor-MAC (S-MAC)

S-MAC [Ye et al., 2002] is a contention-based protocol that regulates sleep periods in a sensor network to conserve energy and improve network lifetime. S-MAC adopts static sleep scheduling for preserving the energy. It divides the time into frames and every frame is divided into an active and a sleep period as shown in Figure 2.8. In active period, the transmitter-receiver is switched on and it is switched off during sleep period. The active period is further divided into Time Synchronization period and data transfer period. Time Synchronization is required so that receiver remains awake when sender sends the message. In Time Synchronization Period, first step in setting the sleep schedule for a node is to listen for a SYNC packet from a neighbour. The SYNC packet contains the sleep schedule and indicates that the sender is going to sleep after "t" seconds. Once the node receives its neighbour's sleeping schedule, it adopts that schedule and re-transmits the schedule for other neighbouring nodes to adopt. Each node within a cluster follows same sleep schedule In S-MAC, data exchange takes place in data transfer period through RTS-CTS-DATA-ACK handshaking for unicast communication. A node may extend its active duration if data exchange doesn't finish in active period. However, even if data exchange finishes within active period, the node will still remain awake until its sleep time and thus, wasting energy.

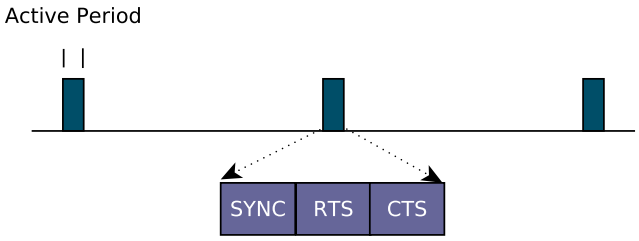


Figure 2.8: S-MAC synchronous periodic wake-up scheme.

Timeout-MAC (T-MAC)

A variation of the S-MAC protocol is the one proposed in [van Dam and Langendoen, 2003] known as T-MAC protocol. This protocol achieves a better performance in terms of energy consumption regarding the S-MAC protocol. Unlike S-MAC, T-MAC follows dynamic sleep schedule. The T-MAC protocol introduces an active timeout mechanism that decreases the idle listening overhead by dynamically adjusting the active period according to network traffic loads. T-MAC allows the nodes to sleep after some time when all network traffic has completed, as shown in Figure 2.9. The end of traffic is signalled after monitoring an idle channel for an adaptive timeout (TA) period. If no activity occurs for this TA time duration, node switches off its radio and goes to sleep state. Such mechanism makes T-MAC more efficient in terms of energy than S-MAC to the detriment of the latency which is increased in T-MAC protocol. This approach is also an improvement regarding the IEEE 802.15.4 protocol where the wake-up scheme is also static and does not change according to the network traffic.

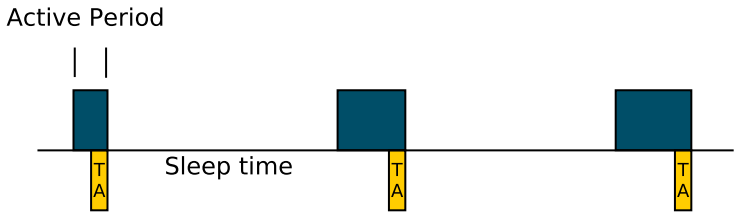


Figure 2.9: T-MAC synchronous periodic wake-up scheme.

2.5.2 Asynchronous Protocols

The disadvantage of synchronous protocols is the fact that they induce additional load and lead to more contentions and collisions by grouping communications in small active periods. Besides, they introduce an overhead due to the synchronisation messages exchanged between nodes. An alternative to this kind of protocol are the asynchronous low-power protocols in which no syncrhonisation is required and therefore reducing the overhead. This is one of the reason why these protocols are widely implemented in many WSN operating systems (TinyOS and ContikiOS). Nevertheless, the communication between nodes are completely decoupled and this issue will have an impact in the delay for the sender to meet the receiver's active period and therefore, affecting the whole end to end delay. Asynchronous protocols rely on *low power listening* (LPL), also called preamble sampling, to link together a sender with data to a receiver who is duty cycling. When a sender has data, the sender transmits a preamble that is at least as long as the sleep period of the receiver. The receiver will wake up, detect the preamble, and stay awake to receive the data. This allows low power communication without the need for explicit synchronization between the nodes. The receiver only wakes up for a short time to sample the medium, thereby limiting idle listening. In the next subsections we give an introduction to the most importants asynchronous protocols.

B-MAC [Polastre et al., 2004] is a variant of the CSMA with preamble sampling mechanism. The preamble sampling is improved with a selective sampling method where only energy above a given noise threshold is considered as useful. This selective measure makes sure that receiver is not wasting its energy just for an insignificant channel activity. When detecting a channel activity over the noise threshold, receiver turn on the radio until packets are received or timeout occurs. CSMA mechanism is used before sending data or preamble packets. However, this protocol requires nodes to remain awake during the duration of the long preamble transmission before obtaining the data packet since there is no information regarding the destination address in preamble packets. In other words, preamble wakes up each node in the neighborhood whether or not it is the target. An schema of the preamble mechanisme in B-MAC is shown in Figure 2.10 where T_i is the whole wake-up period for checking the channel. B-MAC presents basically two disadvantages. First, once a non-target receiver wakes up and detects a preamble, it would have to wait until the end of the preamble to determine that it is not the target and should go back to sleep, introducing thus, an *overhearing problem*. Second, target node has to wait the full period until the preamble is finished before the data/ack exchange, increasing the one hop delay.

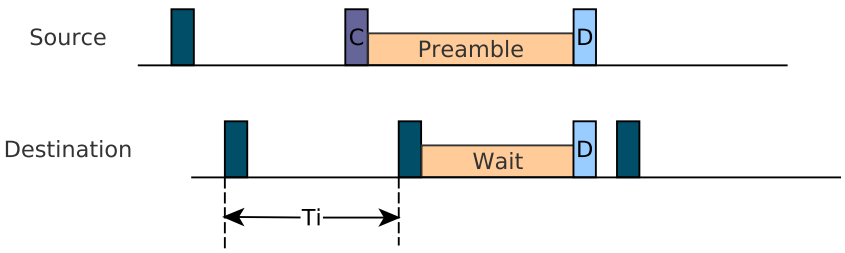


Figure 2.10: Basic operation of B-MAC protocol

X-MAC [Buettner et al., 2006]: X-MAC is an asynchronous protocol that addresses the issues described before, reducing thus the energy consumption and delay. The idea is to use a *short preamble* which embeds information regarding the destination address within the preamble so that the no-target nodes can quickly go back to sleep. In this way, overhearing problem is avoided. To address the second issue where target receiver has to wait until the end of the preamble, X-MAC proposes a *strobed preamble* which allows target receiver to interrupt the long preamble once it determines that it is the target receiver. When a node wakes up and receives a short preamble packet, it looks at the target node ID included in the packet. If the node is not the destination then it goes to sleep mode immediately. If the node is the destination node it sends an early ACK and remains awake for the subsequent data packet. This scheme adresses the overhearing problem and saves the energy consumed by the non-target receivers. Differently from traditional LPL where preamble is sent as a constant stream, X-MAC inserts small pauses between short preambles during the time within which the sender listens to the medium. During this time, receiver can send an *early acknowledgement* packet back to the sender. Once the sender receives an acknowledgement from the receiver it stops sending preambles and sends the data packet (Figure 2.11). Regarding the delay, X-MAC improves the performance in terms of delay over B-MAC since the receiver can inform the sender about its wake-up. In [Beaudaux et al., 2014], authors identify some omissions and black spots not pointed out in the original protocol design and propose alternative and improvements to resolve them.

WiseMAC [El-Hoiydi and Decotignie, 2004]: The idea of this protocol is to use an adaptive preamble in terms of its size. By learning the wake-up period of each neighbor, the preamble size is then reduced to a minimal size. When the wake-up period from neighbors is unknown, the preamble size is equal to the full cycle duration. This mechanism is called *phase-lock*. After a packet is transmitted to the receiver, it sends an acknowledgement adding the information regarding its own schedule. Sender will receive this acknowledgement and will extract the information about neighbor schedule storing this information in a table. In this way, a node can determine the wake-up time of all its neighbors waking up at the exact moment to send data. In other words, the size of the preamble is adapted according to the neighbors's schedule. This approach will further improve the performance in terms of the transmission delay.

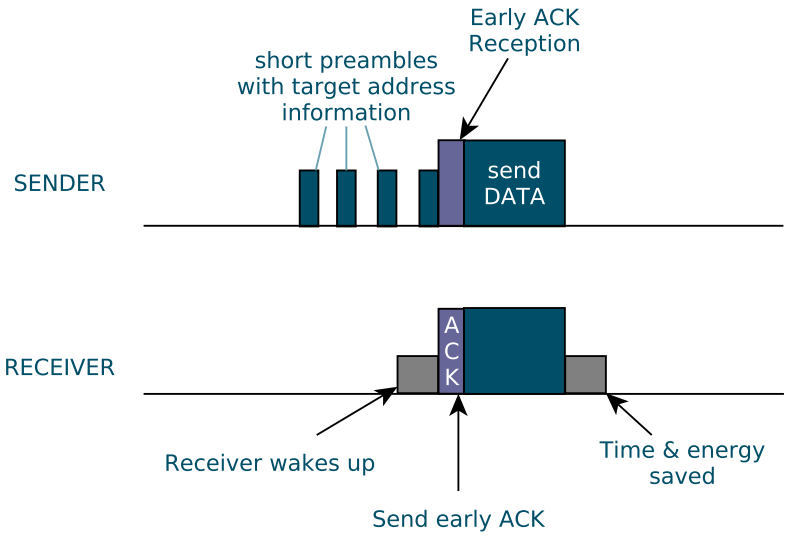


Figure 2.11: X-MAC short preamble approach.

ContikiMAC [Dunkels, 2011]: ContikiMAC protocol is an extension of the X-MAC protocol which includes the phase-lock mechanism proposed in the WiseMAC protocol. ContikiMAC is a radio duty cycling protocol that uses periodical wake-ups to listen for packet retransmissions. If the packet transmission is detected during a wake-up, the receiver is kept on to be able to receive the packet. Receiver sends then the corresponding acknowledgement. To transmit a packet the sender repeatedly sends its packet until it receives the corresponding acknowledgement. This mechanism is ilustrated in Figure 2.12. In the case of broadcast packets, no acknowledgement is

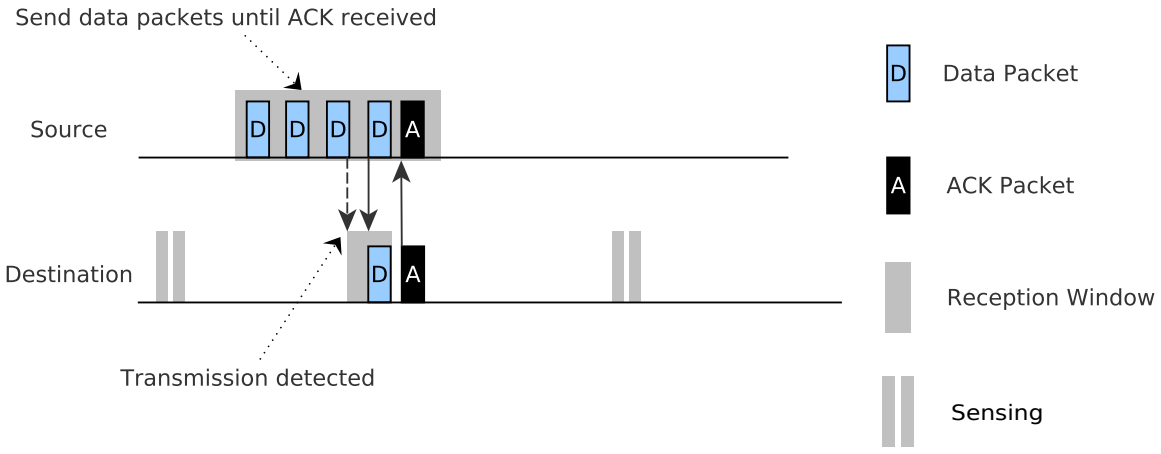


Figure 2.12: ContikiMAC mechanism.

expected. Instead, sender sends data packet all along the full wake-up period. This mechanism is shown in Figure 2.13.

ContikiMAC has a power-efficient wake-up mechanism that relies on precise timing between

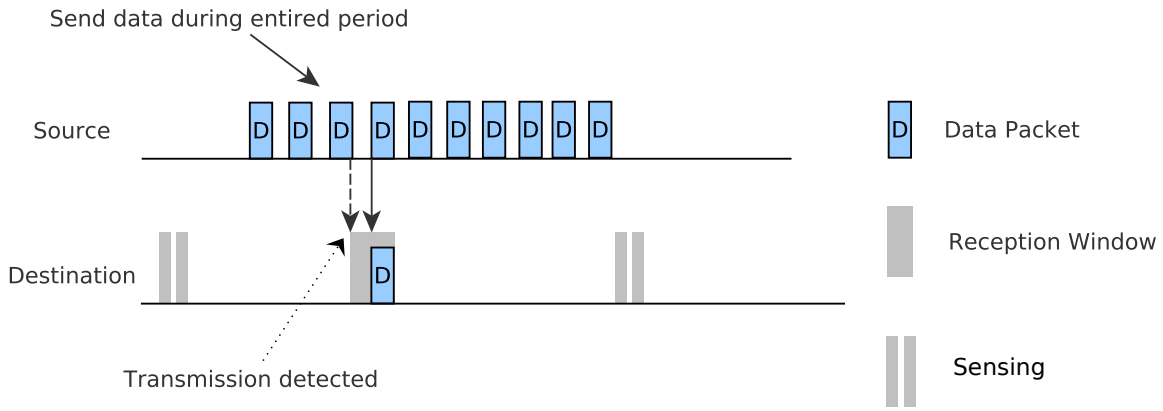


Figure 2.13: ContikiMAC broadcast mechanism.

transmissions. Figure 2.14 shows how timing is handled in ContikiMAC where:

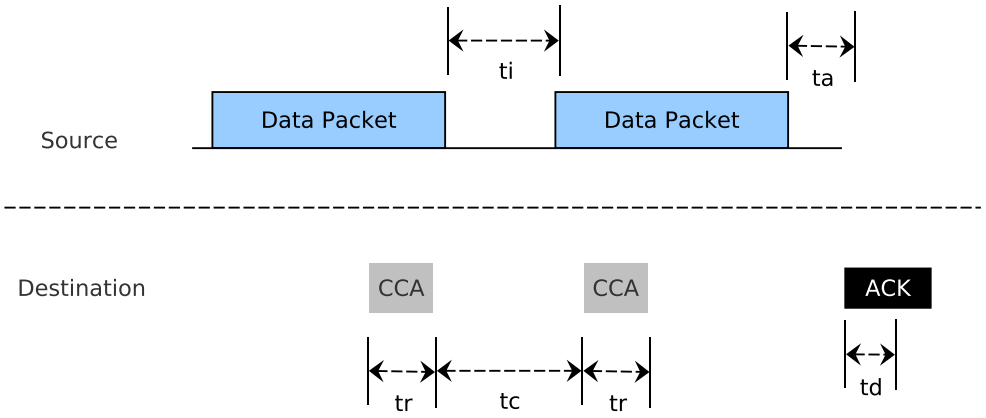


Figure 2.14: ContikiMAC transmission and CCA timing.

- t_i is the interval between transmissions.
- t_r is the CCA interval.
- t_c the interval between each CCA.
- t_a the time between receiving a packet and sending the corresponding acknowledgement.
- t_d the time for successfully detecting the acknowledgement.
- t_l the maximum packet length.
- t_s the shortest packet length. It must satisfy that $t_s > t_r + t_c + t_r$
- t_x the packet size. It must satisfy that $t_s < t_x < t_l$

Timing in ContikiMAC must satisfy the following constraint:

$$t_a + t_d < t_i < t_c < t_c + 2t_r < t_s.$$

ContikiMAC has also a mechanism called *fast sleep* which allows receivers to go sleep earlier if the CCA woke up due to spurious radio noise. This happens if:

- CCA detects activity but the activity is longer than t_l.
- radio activity is followed by a silence period longer than t_i.
- the activity period is followed by a silence period of the correct length followed by activity but no start of packet is detected.

2.6 Overview of RPL protocol

In this section, we give an introduction to a well-known routing protocol for low power and lossy networks: RPL which we have used during this thesis in order to carry out our experiments. RPL [Winter et al., 2012] is a Distance Vector IPv6 routing protocol for LLNs that specifies how to build a Destination Oriented Directed Acyclic Graph (DODAG) using an objective function and a set of metrics/constraints. The objective function operates on a combination of metrics and constraints to compute the "best" path. There could be several objective functions in operation on the same node. For example, several DODAGs may be used with the objective to "Find paths with best ETX (Expected Transmission Count) values" or "Find the best path in terms of latency (metric) while avoiding battery-operated nodes (constraint)". Objective functions then dictate some rules to create the DODAG. A DODAG is a logical routing topology built over a physical network to meet a specific criteria and we can have multiples DODAGs active at the same time. A node then can participate and join one or more DODAGs (RPL instances). The DODAG is created by the root (configured by the system administrator) by an exchange of specific messages. The protocol specification defines a set of messages used to exchange graph-related information. These messages are called DIS (DODAG Information Solicitation), DIO (DODAG Information Object) and DAO (DODAG Destination Advertisement Object).
The root starts by advertising the information about the DODAG using the DIO message. Neighbors will receive the message and will make a decision whether to join the DODAG or not (based on the objective function, local policy, etc). Once the node has joined a DODAG it has a route toward the DODAG root. Root is termed as the "parent" of the node. After computing its rank, it will advertise the DODAG information to its neighboring peers unless the node is a "leaf" in which case it will simply join the DODAG. This process builds the graph edges out from the root to the leaf nodes. In this formation, each node has a routing entry toward its parent in a hop by hop fashion. If a node wants to send a message to the root it should simply forward the message to its immediate parent. This is known as "up" direction traffic. The "rank" is relative and represents an increasing coordinate of the relative position of the node with respect to the root in the DODAG. DIS messages are used by the nodes to proactively solicit graph information from the neighboring nodes. Similar to the "up" direction traffic, it is also necessary for traffic to flow in "down" direction. This requires a routing state to be built at every node and a mechanism to populate these route. This is accomplished by the DAO message which is used to advertise prefix reachability towards the leaf nodes. Once a node joins a DODAG it will send DAO message to its parent set. When parents receive a DAO message it will process the prefix information and adds a routing entry in the routing table. Figure 2.15 shows a DODAG

created by the root node by means of this message exchange mechanism (DIO and DAO). Nodes can solicit graph information by means of DIS messages. Each link has a cost which can be augmented with node related data such as battery, state, etc.
Multiple topology routing can be achieved in RPL thanks to the concept of DODAG instance identified by an instance-id where the idea is to construct multiple graphs over the same physical topology. This gives a way to provide path based on different optimization objectives as specified by the objective function and routing/constraint metrics.
In this thesis we make use of the Contiki implementation of the RPL protocol so two objective functions are taken into account: the best path in terms of the Expected Transmission Count (ETX) and objective function 0, which is basically hop-count.

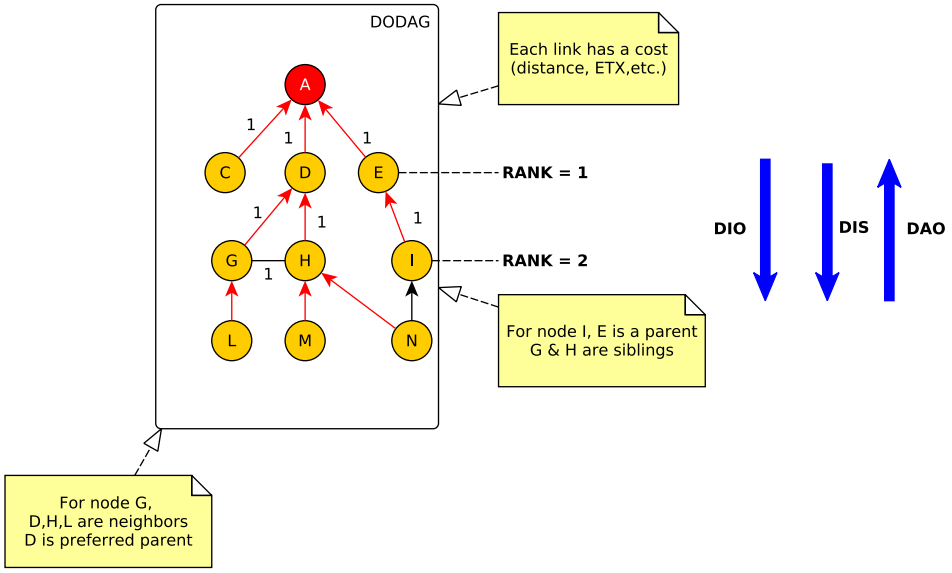


Figure 2.15: RPL DODAGs.

2.7 Conclusion

In this chapter we have presented the necessary preliminaries to understand the problem we are focused on. We introduced the concept of delay explaining the main factors affecting this quality of service metric in a WSN. In this way, we have presented the mechanism of duty cycle, an approach that allows to preserve the energy on each sensor node. Considering the fact that saving energy is one of the most important issues in WSNs, most of the proposed MAC protocols for WSNs are duty cycled. In this way, we have presented the most important low duty cycle MAC protocol commonly used in WSNs, distinguishing between synchronous and asynchronous protocols. Finally, we introduced the well-known RPL routing protocol which is used all along this thesis in order to represent real world scenarios in WSNs for validating our approach.
In the next chapter we give an introduction to typical approaches for estimating the end to end delay in a WSN emphasising the analytical approaches for finding a Markov chain model to model the behavior of the underlying MAC protocol.

Chapter 3

Main Approaches for Estimating End to End Delay in Wireless Sensor Networks

Contents

3.1 Introduction

In this chapter we present the related work concerning the estimation of the end to end delay in WSNs emphasising on the typical analytical approaches found in the literature for modelling the behavior of a WSN, since it is one of the most important issue in this thesis. We also present an analytical framework based on frequency domain analysis for estimating the end to end delay in networks used all along this thesis. We start by presenting an overview of the common approaches followed by the description of the Markov chain approach for modelling the behavior of a WSN. We finish by giving an introduction to the this analytical framework based on frequency domain.

3.2 Typical Approaches for End to End Delay Evaluation

There are basically three approaches for estimating the delay in a WSN: measurements, simulation and analytical models, each of them presenting advantages and disadvantages. We present next related work on each of these techniques.

3.2.1 Measurements

This technique consists in setting up a scenario in a real WSN testbed and to measure the end to end delay from a given node (source) to the destination one. Having ideal time synchronisation component (high precision with negligible overhead), the end to end delay measurement can be simply achieved by taking two timestamps on both source and destination node. However, in a large scale network, global time synchronisation is very costly in both the bandwidth and program memory consumption. Besides, the synchronisation service usually suffers a lot from the unreliable wireless links as well as the environmental interferences. Due to the difficulty to achieve global synchronisation in large-scale sensor networks, measuring the end to end delay is extremely challenging. LiveNet [rong Chen et al., 2007] records the network transmission traces in order to be able to rebuild the network dynamics. However, this method requires additional hardware and its cost is very expensive. Another approach is to keep the per-hop delay records in each data packet. Hence, the end to end delay can be computed by accumulating all per-hop delays once the packet arrives to the destination node. Nevertheless, keeping the timestamps incurs in a non negligible overhead. To overcome this problem, another approach can be to record delay values hop by hop with separate counters. As the large scale sensor networks usually contain long end to end paths, the method may consume a large amount of space in data packets. Another choice is to keep only one counter on each packet that records the accumulated delay. However, such approach is vulnerable to clock skew. In [Liu et al., 2013], authors propose a novel approach named *Sequential Difference Recovery (SDR)* for end to end delay measurement in large scale sensor network without synchronisation. They investigate the key design issues in SDR such as the impact of clock drift and disordered packets. This scheme is implemented on TelosB motes and extensive experiments where done on a testbed of 50 nodes. In this thesis, we propose an approach that combines both measurement and analytical approaches for estimating the end to end delay in a multi-hop transmission scenario. Our work make use of IoT-LAB [Fambon et al., 2014] platform, a large-scale infrastructure suitable for testing and evaluating wireless sensor networks. Several experiments were done all along this thesis using IOTLAB's platform.

3.2.2 Simulation and Emulation

A second technique commonly used to estimate the end to end delay in a WSN is by means of simulation tools. The nature of WSNs makes difficult to analytically model a WSN and predict the actual performance of protocols and network operation. Moreover, deploying and operating a testbed to study the behavior of protocols and network performance supposes a great effort. Consequently, simulation is essential to study WSN, being the common way to test new applications and protocols. The advantage of simulation over measurements is the simplicity for setting up scenarios with a large number of nodes (hundred or thousands), something that is difficult to achieve in real testbeds. Besides, simulations can overcome the problem of clock synchronisation found in measurement approach. However, it requires a suitable model based on several assumptions and an appropriate framework to ease implementation. In addition, simulation results rely on the particular scenario under study, hardware and physical layer assumptions which are not usually accurate enough to capture the real behavior of a WSN. However, simulation is useful for analysing the scalability since we are able to set a large number of nodes for a given scenario. Hence, the tradeoff between scalability and accuracy becomes a major issue when simulating WSNs.

Simulation: Simulation models for WSN include new components not present in classical network simulators such as detailed power and energy consumption models.
NS-2 [Issariyakul and Hossain, 2008] is one of the most popular general network simulators supporting a wide range of protocols in all layers. It provides the most complete support of communication protocol models, among non-commercial packages. Regarding WSN specific protocols, NS-2 includes ad-hoc and WSN specific protocols such as SMAC. OMNET++ [Varga and Hornig, 2008] is a modular discrete event simulator implemented in C++ quite simple to work with due to its clean design. Its major drawback is the lack of available protocols in its library, compared to other simulators.

Simulation & emulation: There are also some specific WSN simulator and emulator tools such as TOSSIM [Levis et al., 2003], Prowler and RMASE [Simon et al., 2003]. TOSSIM simulates the execution of nesC code on a TinyOS/MICA, allowing emulation of actual hardware by mapping hardware interruptions to discrete events. These emulations include the Analog-to-Digital Converter (ADC), the Clock, the transmit strength variable potentiometer, the EEPROM, the boot sequence component, and several of the components in the radio stack. A simulated radio model is also provided. The goal of this simulator is to study the behavior of TinyOS and its applications rather than performance metrics of some new protocols. PROWLER, a probabilistic network simulator, is developed with various radio models and a CSMA MAC layer. RMASE, an application built on the top of PROWLER, provides a layered routing architecture with routing scenario specifications and performance metrics for algorithm evaluations. COOJA [Eriksson et al., 2009] is a simulator and emulator for the Contiki OS for WSNs. It support the typical sensor node platforms, but more important, new devices can be added by users. A survey in simulation tools can be found in [Du et al., 2010]. A useful survey helpful for selecting the most appropriate simulation tool can be found in [Singh et al., 2008].

Application of simulation for WSN performance evaluation: Simulation tools are very useful for evaluating the performance in WSNs. Jurcik et al. [Jurcik et al., 2007] propose OPNET [Lucio et al., 2003], a simulation model for evaluating the GTS mechanism in the standard IEEE 802.15.4 MAC protocol in terms of throughput and delay. They also propose a new methodology to tune the protocol parameters such that a better performance in terms of throughput and delay can be achieved. Authors in [Xie et al., 2014] present a performance evaluation between three routing protocols for WSNs in terms of the average end to end delay by means of COOJA simulation tool. Chaudhary et al. [Chaudhary and Waghmare, 2012] analyse the end to end delay in a WSN using NS-2 simulation for comparing the performance of different routing protocols. Besides, end to end delay is used for comparing two different models: a multiple sink and single sink model.

The main problem of the simulation approach for estimating performance metrics is the fact that models are simplification of the real world. The operating system running on each node is not taken into account, radio channel model are also simplification of the real world. Hence, obtaining reliable conclusions from research based on simulation is not a trivial task.

3.2.3 Analytical Modelling

A third approach for estimating the end to end delay in a WSN is by means of analytical models. This approach consists in finding a mathematical model that fits the behavior of the WSN. Due to the stochastic nature of WSN, the typical approach for modelling them is based on discrete

Markov chains [Norris, 1999] to model the node behavior. Concretely, these Markov chain models focus on modelling the behavior of the underlying MAC protocol. From this Markov chain model, a set of metrics can then be estimated. However, due to the complexity in mathematical terms, most of the proposed models in literature simplifies the reality in order to obtain a treatable mathematical model, leading to not accurate estimations. Besides, existing analytical models were proposed for the IEEE 802.15.4 MAC protocol. Only a few work can be found for other MAC protocols such as S-MAC and X-MAC [Yang and Heinzelman, 2012]. However, model in [Yang and Heinzelman, 2012] is also a simplification since retransmission is not supported, the channel is ideal (no fading, and no capture effect) and there is only one transmission opportunity and one data packet reception per node per cycle.
Most of the theoretical studies are based on the Markov model initially proposed by Bianchi [Bianchi, 2006] for the IEEE 802.11 standard. The model describes the basic functionalities of IEEE 802.11 through a Markov chain under saturated traffic and ideal channel conditions. Extensions of this model have been used to analyze the packet reception rate, the delay [Chatzimisios et al., 2002], [Hadzi-Velkov and Spasenovski, 2003], the medium access control (MAC), layer service time and throughput of IEEE 802.11. A simple and effective analysis of delay distribution is studied for IEEE 802.11 in [Raptis et al., 2006].
Next, we present Bianchi's model together with two extensions of this model found in literature for modelling the behavior of the standard IEEE 802.15.4 MAC protocol. Finally, we present a third Markov model for the standard IEEE 802.15.4 protocol from which authors are able to estimate the end to end delay in a multi-hop transmission scenario.

Bianchi's Model

Bianchi [Bianchi, 2006] proposes a Markov chain model for the standard IEEE 802.11 conceived for wireless networks for local area communications from which he investigates the fundamental mechanism to access the medium called distributed coordination function (DCF). DCF adopts an exponential backoff scheme. At each packet transmission, the backoff time is uniformly chosen in the range $(0, w-1)$. The value w is called contention window and depends on the number of transmissions failed for the packet. At the first transmission attempt, w is set equal to a value CW_{min} called minimum contention window. After each unsuccessful transmission, w is doubled up to a maximum value $CW_{max} = 2^m CW_{MIN}$. The backoff time counter is decremented as long as the channel is sensed idle, "frozen" when a transmission is detected on the channel, and reactivated when the channel is sensed idle again. The station transmits when the backoff time reaches zero. If the transmitting station does not receive the ACK within a specified ACK_Timeout, or it detects the transmission of a different packet on the channel, it reschedules the packet transmission according to the given backoff rules. This mechanism is called the basic access mechanism of DCF.
In his work, author concentrates on the performance evaluation of the DCF scheme, in the assumption of ideal channel conditions and finite number of terminals. DCF describes two techniques to employ for packet transmission. The default scheme consists to send an acknowledgement (ACK) from the destination once it receives a packet from the source. A second optional scheme is a technique known as request-to-send/clear-to-send mechanism. Before transmitting a packet, a station operating in RTS/CTS mode reserves the channel by sending a special RTS frame. The destination station acknowledges the receipt of an RTS by sending back a CTS after which normal packet transmission and ACK response occur. Bianchi's model accounts for all the exponential backoff protocol details and allows to compute the saturation throughput performance of DCF for both access mechanisms.

Proposed Markov Chain Model The analysis focuses in modelling the behavior of a single node with a Markov chain model and they obtain the stationary probability τ that the station transmits a packet in a slot time. This probability does not depend on the access mechanism employed. A fixed number n of contending stations is considered, all of them in saturated conditions, that is to say, each station has immediately a packet available for transmission after the completion of each successful transmission. They consider $b(t)$ as the stochastic process representing the backoff time counter for a given station. A discrete and integer time scale is adopted: t and $t+1$ correspond to the beginning of two consecutive slot times, and the backoff time counter of each station decrements at the beginning of each slot time. A second stochastic process $s(t)$ is defined representing the backoff stage $(0, ..., m)$ of the station at time t, where m is the maximum backoff state such that $CW_{max} = 2^m W$ and $W = CW_{min}$. The notation $W_i = 2^i W$ is adopted for representing each backoff stage, $i \in (0, m)$. The key approximation of this model is that, at each transmission attempt, and regardless of the number of retransmissions suffered, each packet collides with constant and independent probability p. Once independence is assumed and p is supposed to be a constant value, it is possible to model the bidimensional process $\{s(t), b(t)\}$ with the discrete-time Markov chain depicted in Figure 3.1. From this Markov chain,

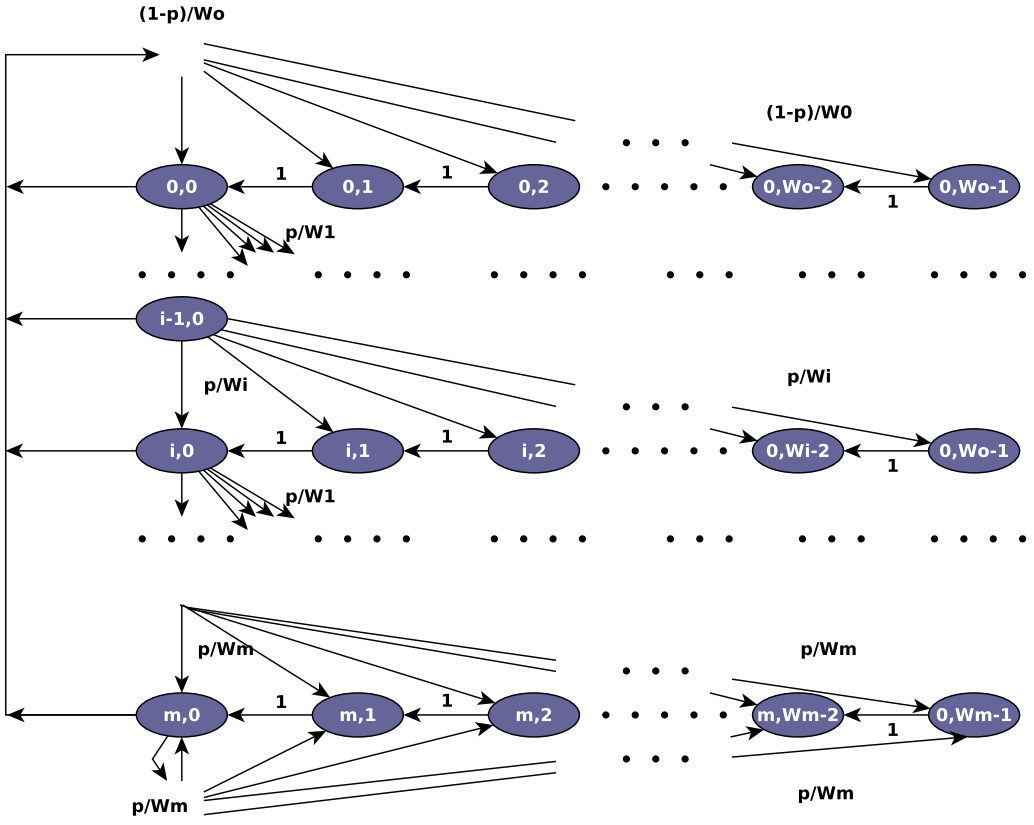


Figure 3.1: Bianchi's Markov chain model for the backoff window size for the standard IEEE 802.11.

the probability τ that a station transmits in a randomly chosen slot time is found by solving a nonlinear system of equations in the two unknown probabilities τ and p. Based on the computed

probabilities τ and p, authors bring an analysis of the throughput for the standard IEEE 802.11 protocol.
In the next section we present two extensions based on Bianchi's model for modelling the behavior of the standard IEEE 802.15.4 MAC protocol in WSNs.

3.3 Modelling IEEE 802.15.4 Slotted-CSMA/CA MAC Protocol

3.3.1 Model of Park et al.

Park et al [Park et al., 2009], propose a Markov chain for modelling the behavior of the slotted version of the standard IEEE 802.15.4 considering retry limits, acknowledgement, unsaturated traffic and a star topology with PAN coordinator and N nodes with beacon-enabled slotted CSMA/CA and ACK. Like Bianchi's model, they define a stochastic process $s(t)$, $c(t)$ and $r(t)$ where $s(t)$ represents the backoff stage at time t, $c(t)$ the state of the backoff counter at time t and $r(t)$ the state of retransmission counter at time t. Under the assumption of independent probability that nodes start sensing, the stationary probability τ that a node attempts a first carrier sensing in a randomly chosen time slot is constant and independent of other nodes, and the tuple $(s(t), c(t), r(t))$ is a three dimensional Markov chain. Authors present results in terms of reliability, delay for a successful received packet, and energy consumption. They also give an approximated delay distribution. The discrete probability distribution function of the delay is approximated by known distributions whose average and variance is matched to the actual average and variance of the delay. However, in this work, delay for a successfully received packet is defined as the time interval from the instant the packet is at the head of its MAC queue and ready to be transmitted, until an ACK for such a packet is received. In other words, queueing delay is not considered in this case; the analysis is limited to a queue of maximum size equals to one which is not a natural scenario in real world WSN. The model also lacks of a real channel model and duty cycle is not taken into account. Since they consider a star topology, results in terms of average delay and distribution of the delay are conceived for one hop scenario and therefore, end to end delay in a multi-hop transmission scenario can not be computed by means of this model. Finally, the model validation was done by means of Montecarlo simulation and thus not considering a real testbed environment.

3.3.2 Model of Misic et al.

As said before, the Markov chain model proposed by Park in [Park et al., 2009] takes some assumptions which do not correspond to real world scenarios. In particulary, duty cycle is not considered in the analytical model. In [Misic and Misic, 2008], authors analyse the peformance of the beacon-enabled version of the standard IEEE 802.15.4 MAC protocol in a star topology with uplink traffic only. Each node is modeled as a M/G/1/K queue system. They analyse the performance of the standard, first by considering a superframe with an active period only, and then by considering also an inactive period (duty cycle). The procedure for obtaining such a model is also based on a stochastic process similar to the one proposed in [Park et al., 2009] except that in this case, deferred transmissions (transmission that cannot be finished within the current superframe) are also considered. Unlike Park's model, Misic's model considers a physical model based on the bit error rate. Then, the probability that the transmitted packet will not collide as well as the probability that the packet is not corrupted by noise are computed. Figure 3.2 shows the Markov chain model of a node executing the slotted CSMA-CA algorithm in a cluster where the superframe has both active an inactive periods.

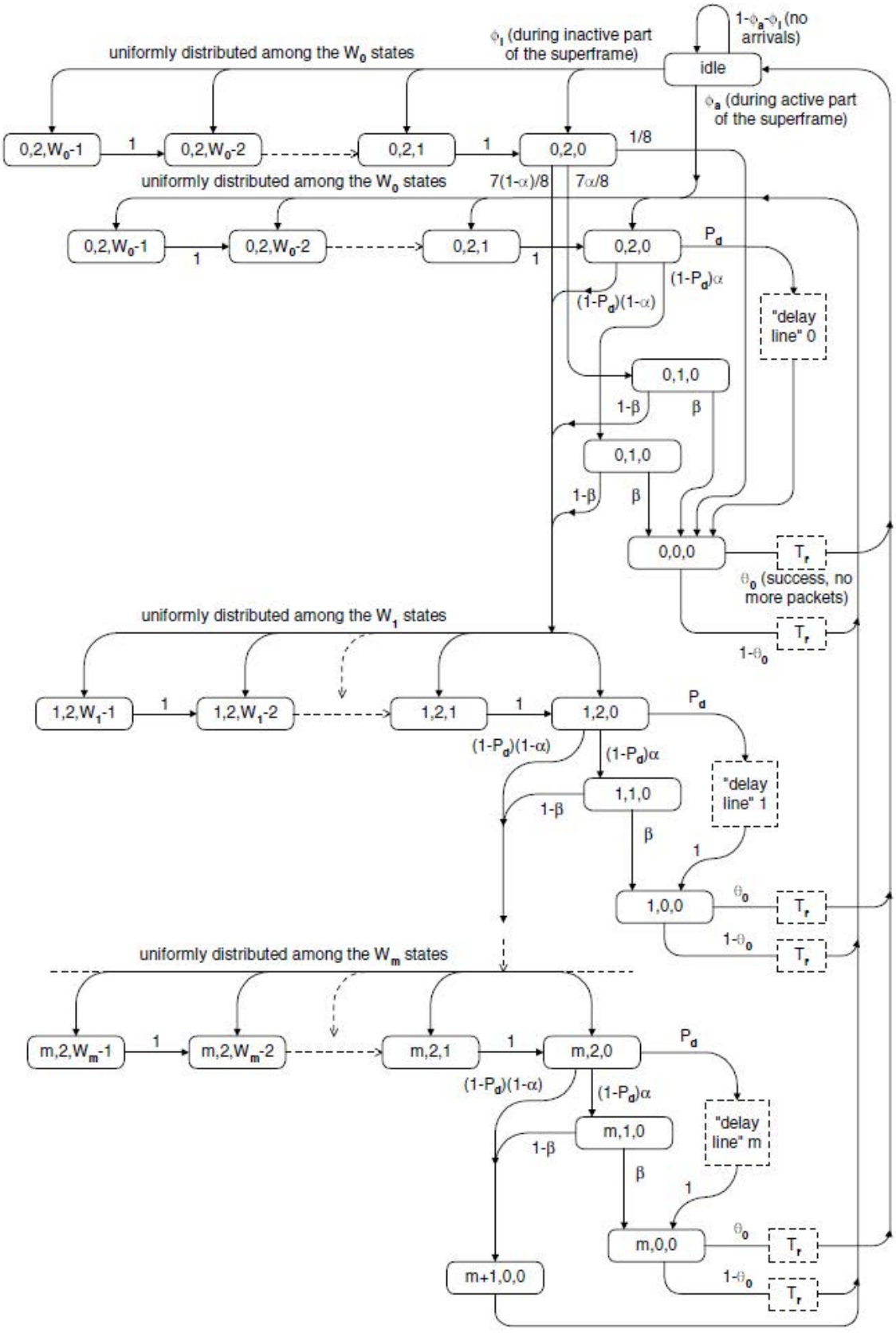


Figure 3.2: Markov chain model of a node executing the slotted CSMA-CA algorithm in a cluster where de superframe has both active and inactive periods [Misic and Misic, 2008]

In this Markov chain, each state has the form $(n(t), c(t), b(t), d(t))$, where

- $n(t)$ represents the value of $NB \in 0..macMaxCSMABackoffs$ - *1*, at time t,
- $c(t)$ represents the value of CW at time t; it may be 0, 1 or 2,
- $b(t)$ represents the value of the backoff time counter which, at the beginning of the backoff countdown, can take any value in the range $0..2^{BE}-1$. During the countdown, the counter decrements at the boundary of each backoff unit
- $d(t)$ representing the current value of the delay line counter (started if packet is deffered) at time t. period. The value $b(t)$ will be frozen during the inactive portion of the beacon interval, and countdown will resume when the next superframe begins.

The non-null transition probabilities define a set of equations from which a set of probabilities can be found, in particulary, the probability α that the channel is sensed idle at the first CCA check, the probability β that the channel is sensed to be idle at the second CCA check, the probability τ to access the medium, the probability of success transmission γ, among others. A system of equations together with the computed probabilities allow to compute a set of performance parameters, in particulary,

- **The probability distribution of the queue length**
  - Queue length at the time of packet departures.
  - Queue length at arbitrary time.
- **The access delay**, that is to say, the interval from the packet arrival at the device queue until the successful packet departure from that queue. A successful packet departure is defined as the packet transmission which is successfully acknowledged.
  - Average delay
  - Probability distribution of the access delay.

This Markov chain model is more complete regarding the one proposed by [Park et al., 2009] in the sense that many aspects found in WSNs are taken into account, for instance, a radio model, duty cycle, queue buffer on each node, etc. However, as mentioned before, this model considers a star topology. Hence, the delay distribution is found in a one-hop fashion. Considering that they found a mathematical expression for the one-hop delay, why not to extend this model in order to consider a multi-hop transmission scenario ? In other words, the same analysis can be done for each node belonging to the network and thus it should be a way for finding a mathematical expression for the end to end delay distribution. The problem here is that the analysis done in this work considers each node as a M/G/1/K queue system. Even if we consider the first node as an M/G/1/K queue, we do not know the distribution of each packet leaving this node and therefore, the arrival flow to the next node is not necessarily Poisson. Hence, instead of

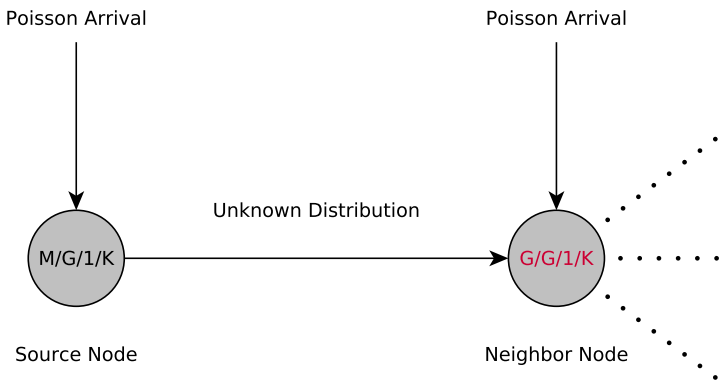


Figure 3.3: Limitation of Misic model for modelling intermediate nodes.

modelling the second node as M/G/1/K, it should be modeled as a G/G/1/K where the input flow is unknown. This issue is shown in Figure 3.3. Therefore, extending the proposed model for estimating the end to end delay in a multi-hop transmission scenario is not possible. Another important point of this model is the fact that it considers a simple radio model which does not represent the real world in WSNs. Experimental studies have shown that real deployments have a *transitional region* with highly unreliable links and therefore, the idealized disc model typically considered can be misleading. These studies have revealed the existence of three distinct reception regions in a wireless link: connected, transitional, and disconnected. Authors in [Zuniga and Krishnamachari, 2004], proposed an analytical model based on the log-normal shadowing path loss model. By taking into account the importance that highly unreliable links in the *transitional region* have on upper-layer protocols, they identify and quantify the influence of this region in the performance of the network. Expressions derived show how *transitional region* is impacted by radio parameters, frame size as well as important environmental parameters such as path loss exponent and the log-normal shadow variance.

3.3.3 Cross-Layer Modelling

Authors in [Wang et al., 2012] present a comprehensive and accurate cross-layer analysis framework to characterize the end to end delay distribution in WSNs for both deterministic and random deployments of nodes. This approach is similar to real-time queueing theory [Lehoczky, 1997] in that they use a stochastic queuing model for the analysis. Their analyse is focalised in finding the probability distribution function (pdf) of single-hop delay between two nodes i and j for both deterministic and random deployment. Then, given the single-hop delay, they estimate the pdf of the end to end delay between the source node i and the destination one s. In this work, each node is modeled according to a discrete-time *Geom/PH/1/M* queuing system. The communication system at each node is modeled as a discrete-time recurrent Markov chain $\{X_n\}$. This Markov chain has a layered structure as shown in Figure 3.4 (a) where each layer i contains the part of the chain where there are i packets in the queue. Transitions among states in $\{X_n\}$ represent the communication behaviors of each node. A second Markov chain $\{Y_n\}$ shown in Figure 3.4 (b) which is the absorbing variant of $\{X_n\}$, is considered for obtaining the single-hop delay distribution. $\{X_n\}$ is composed of χ blocks, denoted as $\{Z_n\}$, each one modelling a single

transmission attempt and depends on the MAC protocol employed in the system. Packets are dropped if they arrive at a full queue or if all χ transmission attempts fail. Then, the equilibrium state distribution of the system can be derived using $\{X_n\}$. Before a packet arrives, the system is in one of the states according to the equilibrium state probability vector of $\{X_n\}$. Using this vector, the initial state probability vector for $\{Y_n\}$ is found. To derive the single-hop delay distribution, the absorption time of the Markov chain $\{Y_n\}$ is considered. With each hop modeled as a Geom/PH/1/M queue, the entire network is considered a queueing network. The end to end delay is then obtained using an iterative procedure.

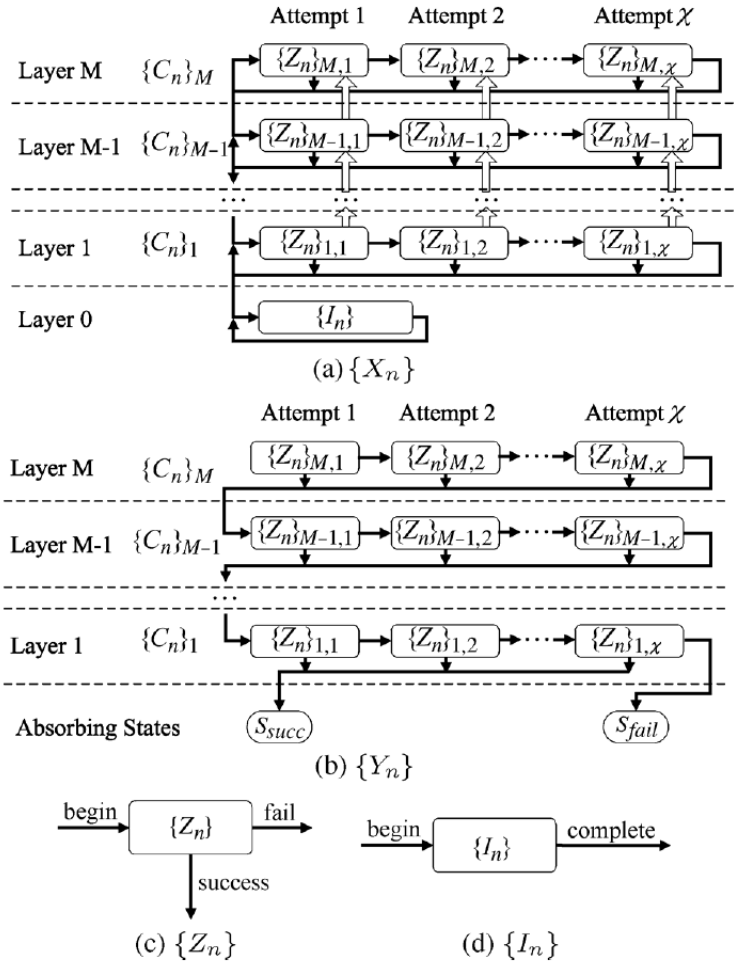


Figure 3.4: Cross-layer Markov chain model to estimate the one-hop delai in one node [Wang et al., 2012].

However, authors consider the same model for all nodes in the network, even for relay nodes. In other words, packet arrival distribution is considered as geometric for all nodes. Authors have performed empirical measurements in order to estimate the relay traffic, that is to say, the traffic arriving to intermediate nodes. Based on these experiments, authors conclude that the geometric distribution closely models the inter-arrival rate. However, these experiments do not consider bursty traffic, a characteristic which is present in some WSN applications such as surveillance

and habitat monitoring applications. Besides, results are subordinated to a specific CSMA MAC protocol used in the experiments. On the other hand, we have seen that the main problem of Misic's model is that we are not able to extend this model to estimate the end to end delay since intermediate nodes can not be modeled in the same way as the source node. This is due to the fact that we are not able to characterize the arrival packet distribution. Authors then, overcome this difficulty by approximating the packet arrival distribution of intermediate nodes to the detriment of a realistic model representing the real behavior of these nodes.

3.4 Frequency domain Analysis for end to end delay Estimation

Some recent work tried to better capture the WSN behavior by providing different analytic models. In [M. Xie, 2009], by considering a linear network (representing a path in the WSN), the e2e delay is analyzed using queueing theory for TDMA and slotted ALOHA MAC protocols. Each node is modeled by a GI/Geo/1 queue and the dependance between neighboring nodes is characterized by a correlation parameter. This result is, however, difficult to be used for analyzing general WSN. More recently, in [He et al., 2010a], authors presented a more general theoretic framework based on frequency domain analysis (i.e. generating function like Laplace transform) for estimating e2e delay distribution within a multi-hop network. End to end delay distribution is usually analyzed in the time domain using mainly Markov models. However, authors in [He et al., 2010a] analyse the end to end delay in a networked system using frequency-domain modelling and analysis, where they have shown that their approach is more scalable and allows analysis of compositional networked systems. They convert the analysis from time domain to frequency domain based on *Laplace Transform* in order to avoid computing the convolution of individual blocks. After getting the analysis results in frequency domain they convert the results back to time domain by means of the *Inverse Laplace Transform*. This approach for computing the delay distribution converts the convolution calculation to multiplication, simplifying the calculation greatly. The idea consists in representing a network as a graph with a set of nodes and links between them and then to assign the reliability function, which could be tipically the packet delay probability distribution, to each of the components (nodes and links) in the network. They define four common building blocks and they analyze the distribution of each of them in order to take them as a start point for calculating the delay distribution in more complex scenarios: a serial structure, a parallel structure, circular structure and a back-up structure shown in Figures 3.5 and 3.6. where $v_j(s)$ represents the delay function of a node and $e_{ij}(s)$ the delay function of a link in frequency domain. As we are working in frequency domain, the expressions of the end to end delay can be found as follows: for the serial structure, the end to end delay is the product of all delays on individual components or links from v_1 to v_n.

$$e_{1n}(s) = \prod e_{ij}(s) = e_{12}(s) \cdot e_{23}(s) \cdots e_{(n-1)n}(s) \tag{3.1}$$

In the parallel structure, several alternative paths exist from a source to a destination. A path e_{ijk} from v_i to v_j is selected with a certain probability p_k. The expression then for the end to end delay in a parallel structure can be found as follows:

$$\begin{aligned} e_{ij}(s) &= \sum_{k=1}^{n} p_k \cdot e_{ijk}(s) \\ &= p_1 \cdot e_{ij1}(s) + p_2 \cdot e_{ij2}(s) + \cdots + p_n \cdot e_{ijn}(s) \end{aligned} \tag{3.2}$$

The circular structure contain two nodes $v_i(s)$ and $v_j(s)$, and two links $e_{ij}(s)$ and $e_{ii}(s)$ where $e_{ii}(s)$ is a feedback loop. To get the closed form of the end to end delay probability density

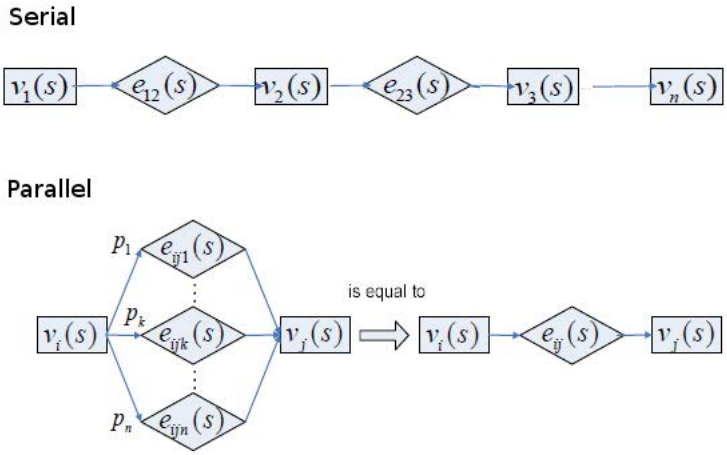


Figure 3.5: Serial and parallel structures [He et al., 2010a].

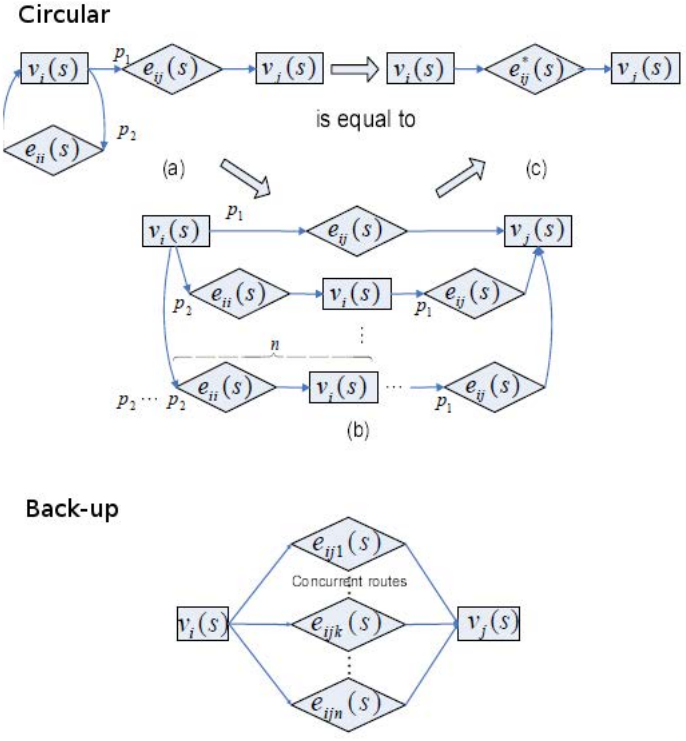


Figure 3.6: Circular and back-up structures [He et al., 2010a].

function of the circular structure ($e^*_{ij}(s)$) in the Figure 3.6(c), they use the block shown in (b) to approximate the circular structure. After some calculus, the expression for the circular structure

is:

$$e^*_{ij}(s) = \frac{p_1 \cdot e_{ij}(s)}{1 - p_2 \cdot e_{ii}(s)} \tag{3.3}$$

Finally, the backoff structure includes a source node $v_i(s)$, a destination node $v_j(s)$, and links $e_{ij1}(s), e_{ij2}(s), \cdots, e_{ijn}(s)$. This structure differs from the circular one because there are concurrent instead of alternative paths. This structure is useful for modelling broadcasting delivery messages. Supposing that $e_{ijk}(s)$, $k = 1, 2, \cdots, n$ are independent of each other, the end to end distribution function of the back-up structure $e_{ij}(s)$ as follows:

$$e_{ij}(s) = min\{e_{ij1}(s), e_{ij2}(s), \cdots, e_{ijn}(s)\} \tag{3.4}$$

Complex scenarios are then represented by one or a combination of these building blocks. A summary of the notation used by the calculus framework is presented in Table 3.1. Once a scenario is specified they proceed to calculate the system e2e delay distribution by means of the *Reliability Adjacency Matrix*. This matrix represents a one hop delay relation of any pair of nodes. Thus, as shown in [He et al., 2010a], for any nodes v_i, v_j:

$$A_{i,j} = \begin{cases} 0 & \text{no direct connection} \\ p_{ij}e_{ij}(s) & \text{direct connection} \\ p_{ik}e_{ik}(s) \cdot p_{kj}e_{kj}(s), & \text{serial structure;} \\ \sum_k p_{ijk}e_{ijk}(s), & \text{k-parallel structure;} \\ \frac{p_{ij}e_{ij}(s)}{1-p_{ii}e_{ii}(s)}, & \text{circular structure;} \\ min\{e_{ij1}(s), e_{ij2}(s), \cdots, e_{ijk}(s)\}, & \text{k-backup structure;} \end{cases}$$

Then, they define matrix operations according to the network topology and the nature of the

Notation	Meaning
e_{ij}	a direct link between v_i and v_j
p_{ij}	the probability of selecting the link e_{ij} for going from v_i to v_j
e_{ijk}	set of $k-$parallel links from node v_i to v_j
p_{ijk}	in a parallel structure means the probability of choosing the e_{ijk} link

Table 3.1: Table of notations of the reliability calculus framework.

protocols. In this way, they define the *Reliability Adjacency Matrix* in r hops as A^r which describes the e2e communication delay distribution along all links between any two components in the given system within r hops. Examples on this can be found in [He et al., 2010a]. The *Reliability Adjacency Matrix* allows to get the frequency-domain expression of the end to end communication function which is then converted to a *general form* expression. Finally, they apply the *Inverse Laplace Transform* to get the end to end communication delay distribution function in time domain. They also provide a hierarchical approach to get the e2e delay distribution in large-scale networks. Even though this calculus framework proposes a set of building blocks that can be used as a starting point to compute the delay in more complex scenarios, this task is still difficult, mainly because of the following factors. 1) it is in general very hard to obtain the delay distribution function of a network node; 2) the theoretic framework assumes the independence of the distributions of the neighboring nodes, which is not verified in general; 3) the queue stability condition has not been explicitly given in [He et al., 2010a], but it is obviously necessary.

3.5 Conclusion

In this chapter we have presented an overview of the approaches found in literature for estimating the end to end delay in WSNs. We have seen that simulation models are normally simplification of the real world and therefore, this approach is not accurate enough for estimating performance parameters. We have also seen that, due to the difficulty to achieve global synchronisation in large-scale sensor networks, the measuring approach for estimating the end to end delay is extremely challenging. Analytical solutions found in literature for modelling the behavior of the MAC layer protocol for estimating the local delay within a node, were also presented. We have seen that, due to the complexity in mathematical terms for modelling the behavior of MAC protocols in real conditions, most of these analytical approaches simplifies the reality by considering ideal channel conditions, by simplifying the queue model on each node, or even not considering the issues introduced by the underlying operating system. Besides, most of the proposed analytical solutions are conceived for the standard IEEE 802.15.14 MAC protocol and considering a star topology. Hence, the whole delay is estimated in terms of the one-hop delay from a source node to the coordinator located at the center of the topology. The extension of these analytical models for considering a multi-hop topology is not trivial and therefore the delay estimation in a multi-hop transmission scenario is not possible. Finally, we have presented a theoretical framework based on frequency domain for estimating the delay. Further, we make use of this framework for estimating, first, the local delay in a node, and then we extend this analysis for estimating the end to end delay.
We have seen that, one of the problems of the analytical approaches is the complexity to capture all the features in a WSN. As we will see further in this thesis, the operating system running on each node introduce non negligible delays that impact in the whole one hop delay which are not taken into account in analytical models leading then to wrong estimations of the delay. On the other hand, and since measures are taken during the execution of a WSN, the measurement approach is suitable for capturing all features in a WSN. Our idea then, is to take advantage of the measurement approach in order to capture the main features of the WSN execution by combining it with an analytical approach modelling the behavior of each of the nodes. This analytical model will allow us to estimate the delay distribution in one hop fashion for each of the node. Then, by means of the frequency domain approach presented before, we can extend this analysis for estimating the distribution of the end to end delay in a multi-hop transmission scenario. In next chapter, we validate results obtained by the proposed approach regarding e2e delay distribution. These results are compared with queueing analysis and simulations by means of a two concrete scenarios. We have applied a statistical analysis on data sets obtained from the different approaches to obtain the Cumulative Density Function (CDF) of the e2e delay to quantify the probabilistic QoS guarantee of the network.

Chapter 4

Experimental Analysis of One-hop Delay in Wireless Sensor Networks

Contents

4.1 Introduction

In this chapter, we present some preliminary results regarding the analysis of the one-hop delay in WSN that we have obtained during this thesis. As we mentioned in previous chapter, during this thesis, we make use of an existing analytical framework based on frequency domain [He et al., 2010a] for estimating the end to end delay. However, some important issues such as the queueuing delay, were not taken into account in this framework. The first preliminary result concerns an empirical validation of this theoretical framework when considering queueing delay in nodes for the case of low and moderate traffic. Next, and considering the fact that most commonly used MAC protocols implement the duty cycle mechanism in order to save energy, we present some measurement results quantifying the impact of this mechanism in a WSN and

its effects in both delay and packet reception rate. We have also considered the analytic model proposed by Misic et al. for modelling the behavior of a WSN in order to be able to estimate a set of performance parameters. This model seems to be accurate for modelling a WSN since most important factors such a duty cycle are taken into account. In this direction, we have done a testbed measurement experiment in a real testbed and we have compared results with the ones obtained by Misic's model in order to determine the accuracy of this for estimating the delay in a WSN.
Since Misic's model focuses in the IEEE 802.15.4 protocol, both testbed results were done by considering the TKN154 implementation of the protocol over TinyOS. An overview of the TKN154 implementation is also presented.

4.2 Extending Frequency Domain Analysis Approach

In [He et al., 2010a], authors propose a way to compute the end to end delay in a set of predefined building blocks. However, none of the provided examples show how to deal with queueing delay in nodes and how this delay influences in the e2e delay computation. A widely used method to analyse communication delays is queueing networks based modelling [Allen, 1990]. This method provides a probabilistic estimation of the e2e delays regarding an arrival model of messages and a service time model to process these messages by nodes. We introduce two scenarios in which we take into account, firstly, the queueing delay in nodes within a Jackson Network [Allen, 1990] where Jackson's results were applied, and then the queueing delay when considering a MAC layer protocol.
Jackson's Theorem states that in a queueing network having K nodes and satisfying that

- Each node consists of c_i identical exponential servers with service rate μ_i
- External messages arrive from outside at queue i in a Poisson pattern with rate γ_i
- Once message is served in queue i it goes to queue $j \in 1, 2, ..., K$ with probability p_i or leave the network with probability $1 - \sum_{j=1}^{K} p_{ij}$

then the average arrival rates $\lambda = (\lambda_1, \lambda_2, ..., \lambda_K)$ to each queue can be computed as

$$\lambda = \gamma(I - P)^{-1} \tag{4.1}$$

where matrix I is the identity matrix, P is the transition matrix from one queue to another in terms of probability, i.e. P_{ij} represents the probability of transition from queue i to queue j and vector γ represents the external message arrival rate for each queue i. Furthermore, each queue i behaves as if it is an independent $M/M/c_i$ queueing system with average arrival rate λ_i. With these ideas in mind we have analysed the queueing delay in a Jackson network and we have computed the entire e2e delay by merging the framework calculus together with these queueing delay results.

4.2.1 Comparative Evaluation

Methodology

In order to verify the suitability of the theoretical approach presented in [He et al., 2010a] (called hereafter *framework*) we have defined two empirical experiments using two simulator tools: JMT,

which is a performance evaluation suite based on queueing network models, and Prowler which runs under MATLAB and provides an easy way of application prototyping with nice visualization capabilities. We have defined a Jackson's scenario which we have tested using JMT and we have compared the results with the ones obtained by the framework. The scenario is a linear network with nodes *in tandem* having certain arrival rate γ in which nodes or queues introduce a delay component due to the queueing delay before transmitting the messages toward the next node. A description and results for this scenario are shown in next section. We have also defined a network scenario in which our purpose is to compare the approach described in the framework [He et al., 2010a] with an empirical example, that is to say, to show how close to a more realistic scenario the framework is. In this way, we have simulated the proposed scenario in Prowler simulator for two different cases of the workload: ρ (0.1, 0.5). Our idea is to use Prowler in order to take advantage of its layered architecture to introduce a MAC layer which is based on CSMA/CA. We have analysed the behavior of this protocol and we then estimated the delay distribution function of links and nodes, which are the building blocks to use within the framework. The network topology is the same as the one we used for the JMT scenario but with different parameters. In order to compare the results we have plotted the results for both Prowler and framework considering different values of ρ. Next subsections describe in detail the evaluation results and the analysis we have done.

Results

We show the results obtained by running the scenarios in JMT and Prowler and we compare them with the theoretical results obtained in [He et al., 2010a]. Both theoretical and empirical delay distributions are shown. For both scenarios, based on the *kernel density estimation* [Parzen, 1962], we have estimated the probability density function of the empirical collected data and plotted the corresponding CDF together with the theoretical CDF.

JMT Scenario In order to show how the framework behaves when dealing with queueing delay we have included a Jackson Network [Allen, 1990] scenario which we have ran using JMT simulation tool. Based on the Jackson's analysis shown in previous section and considering that queueing delay is not negligible and must be taken into account when analysing the local and entire delay, we introduce as a contribution, a queueing based analysis, which helped us in determining the local delay distribution, and the implications of adding it to the framework. We have applied Jackson's results explained before. Figure 4.1 shows the scenario we have analysed. Nodes delay have exponential distribution with parameter $\mu = 34$ while links delay

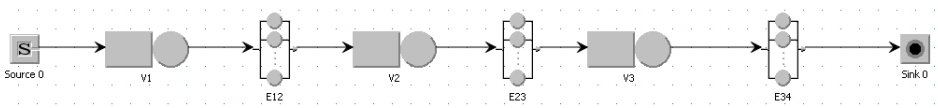


Figure 4.1: Open queueing Jackson network based scenario.

have exponential distribution with parameter $\mu = 1$. Messages arrive as Poisson process with parameter $\gamma = 17$. We determined the local delay of each node as follows:

Since there are no feedback transitions, the matrix P is

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Then, the γ vector is (17 0 0). By applying equation 4.1 we obtained a λ-vector (17 17 17) which specifies the arrival rate λ_i of messages for all the i queues in the system. We then compute the sojourn time distribution for each queue which is exponentially distributed with parameter $\mu(1-\rho)$ where $\rho = \frac{\lambda}{\mu}$.
Once we have computed the nodes delay distribution $v_i(s)$ and the links delay distribution $e_{ij}(s)$, we computed the e2e delay by applying the serial formule from [He et al., 2010a] as follows:

$$\begin{aligned} e_{src,sk}(s) &= \prod v_i(s) \cdot e_{ij}(s) \\ &= v_1(s) \cdot e_{12}(s) \cdot v_2(s) \cdot e_{23}(s) \cdot v_3(s) \cdot e_{34}(s) \end{aligned} \tag{4.2}$$

Results are shown in Figure 4.2.

CSMA/CA scenario In order to provide a more realistic scenario we have defined a scenario based on an *in tandem* structure where we incorporate the effect of a MAC layer on each node. We have voluntarily chosen a simple scenario in order to compare both approaches. However, in [He et al., 2010a], a set of network building blocks were defined and can be used to model more complex network topologies. In this scenario our idea is to incorporate attributes from real networks and specially regarding the MAC layer in which delay is affected by many parameters such as random backoff time, busy channels, collision errors, among others. Scenarios considered in [He et al., 2010a] do not consider any MAC layer so we introduce this one in order to see how the e2e delay behaves in a simulator which emulates a real network system. We have used the *CSMA/CA* protocol implementation provided by the Prowler simulator. Messages arrive at node 1 as Poisson Process with parameter γ and travels through node 2 and 3, which are relay nodes, until reaching the fourth and final node. Neighbor nodes are separated by a distance of 0.5 meters. The radio propagation model is shown below

$$P_{rec,ideal}(d) = P_{transmit}\frac{1}{1+d^{\tau}} \tag{4.3}$$

The fading effect is modeled by random disturbances in the simulator. The received signal strength from node j to node i is calculated from the propagation function by modulating it with random functions

$$P_{rec}(i,j) = P_{rec,ideal}(d_{ij}) \cdot (1+\alpha(d_{ij})) \cdot (1+\beta(t)) \tag{4.4}$$

A parameter p_{error} determines the probability that a collision occurs when a message is sent from one node to another. Collisions also happen in the receiver side if a message is sent to it and it has already started to receive another message. In case a collision occurs, the message is dropped. We have set $p_{error} = 0$ so collision may only occurs in the reception process. The channel bitrate is 40Kbps. Finally, the maximum queue length for each node is 20. As we will see further, this queue length is enough to guarantee that no messages are dropped due to full queues. The maximum number of attempts in backoff mode is 10. Messages in backoff that reach this threshold are dropped. We proceed now to compute the estimated service time of each node considering the parameters that we have previously defined. In particular, the estimated service

Notation	Meaning
$P_{rec,ideal}(d)$	ideal reception signal strength.
$P_{transmit}$	transmission signal power.
d	distance between transmitter i and the receiver j.
τ	decay parameter, with value 2 in our case.
$P_{rec}(i,j)$	real reception signal strength.
α	random variable $N(0,\sigma_\alpha)$. Distance dependent.
β	random variable $N(0,\sigma_\beta)$. Time dependent.
σ_α	0.45
σ_β	0.02
waiting time	5ms + 3.2ms * U(0,1)
backoff time	2.5ms + 0.7ms * U(0,1)
max queue length	20
max backoff attempts	10

Table 4.1: Table of notations of the propagation and fading models.

time depends on the packet size, channel bit rate and also the waiting and backoff times. The real delay for sending a packet is:

$$Delay_{pk} = p(waiting\ time + packetTime) + (1-p)(backoff\ time)^k \tag{4.5}$$

where k is the number of backoff which has geometric distribution, p is the probability that a node finds the medium idle when it attemps to transmit and *packetTime* is the time taken for the channel to send all packet bits. Since the packet length is *960 bits* and having a channel bitrate of *40Kbps*, the time taken for a node to transmit a packet is *packetTime=24ms* (960/40Kbps). Since we don't have an estimation of parameter p we have taken a more simplistic assumption when modelling the node delay. Considering that the delay in 4.5 has some constant components we have taken this constant to model the links delay. We have taken the *packetTime* and the constant part of both *waiting time* and *backoff time* as the parameter to model the links having constant distribution with parameter $m = 0.029ms$ ($5ms+2,5ms+24ms$). Nodes were modeled with exponential service distribution as well but taking the random part of *waiting time* and *backoff time*. In this case, the parameter $\mu = 256$ ($=\frac{1}{3.2ms+0.7ms}$). Since it is also an *in tandem* scenario we consider the serial structure formule 4.2 for calculating the e2e delay distribution. We have tested this scenario taking two different parameters of ρ, namely $\rho = 0.1$ and $\rho = 0.5$ as shown in Figures 4.3 and 4.4. Considering that messages arrive as Poisson process and since nodes has exponential service time, we have modelled them as a M/M/1 queue. In this case, the Laplace transform of the delay in nodes is:

$$Delay_v = \frac{\mu - \gamma}{(s + \mu - \gamma)} \tag{4.6}$$

On the other hand, we have considered constant delay in links. Taking the assumption that the output rate of M/M/1 is also Poisson, we have modelled the links as a M/D/1 queue where the service time has constant parameter $\mu = \frac{1}{5ms+2,5ms+24ms}$. In this case, the Laplace transform for the links delay is

$$Delay_e = \frac{s(1-\rho)e^{-sm}}{s - \gamma + \gamma e^{-sm}} \tag{4.7}$$

where $\rho = \frac{\gamma}{\mu}$. We have mentioned that during the simulation there are messages that were dropped. These messages are those that reach the maximum number of attempts (backoff threshold parameter) to find the channel idle. That means that the real arrival rate to each queue depends on the number of dropped messages. The effective arrival rates for the case $\rho = 0.5$, together with some other results, are summarized in Table 4.2. In the case of $\rho = 0.1$ there are no drops so the parameter γ in this case is the same for all queues. The first column

Queue	% Drops	% Drops Full Queue	γ_i	Avg. Backoff
1	12	0	17	5.1
2	22	0	14.9	5.6
3	12	0	11.7	4.9

Table 4.2: Simulation Summary.

in Table 4.2 represents the queue number, the second column shows percentage of messages dropped, the third one the percentage of messages dropped due to full queues, γ_i is the effective arrival rate to each node and the last column shows the average number of attempts to find the channel idle. As we can see, the third column confirms that the queue length parameter shown in Table 4.1 is enough to consider no message loss due to full queues. By applying 4.6 and 4.7 with the corresponding γ and μ parameters we proceeded to compute the delay of each node and link in the system. Then, e2e delay holds by applying 4.2. A discussion of the results is presented in next section.

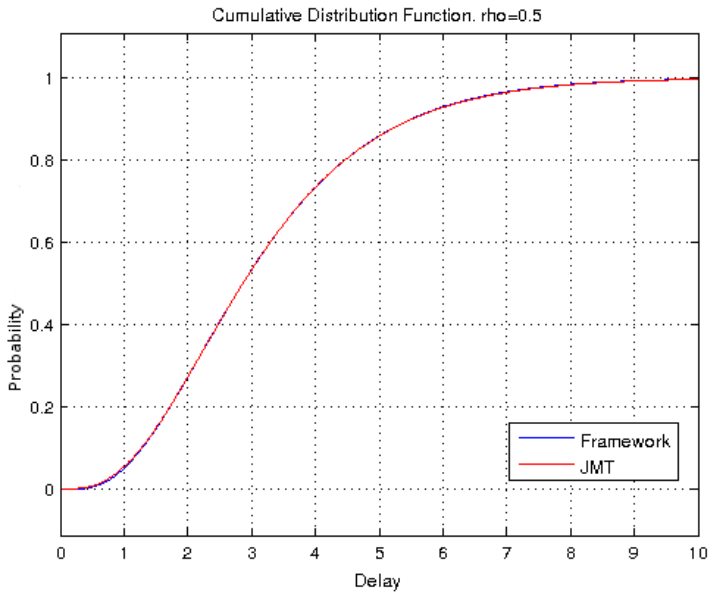


Figure 4.2: Comparison between Framework and JMT with $\rho = 0.5$.

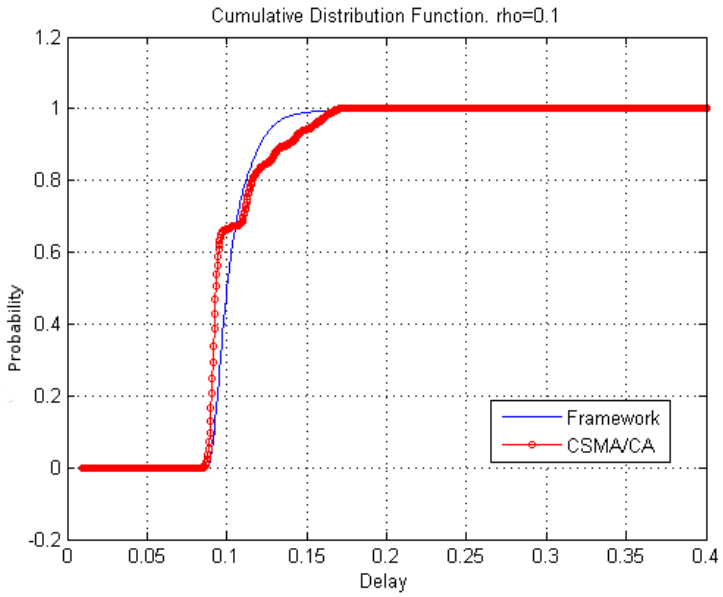


Figure 4.3: Comparison between Framework and CSMA/CA with $\rho = 0.1$.

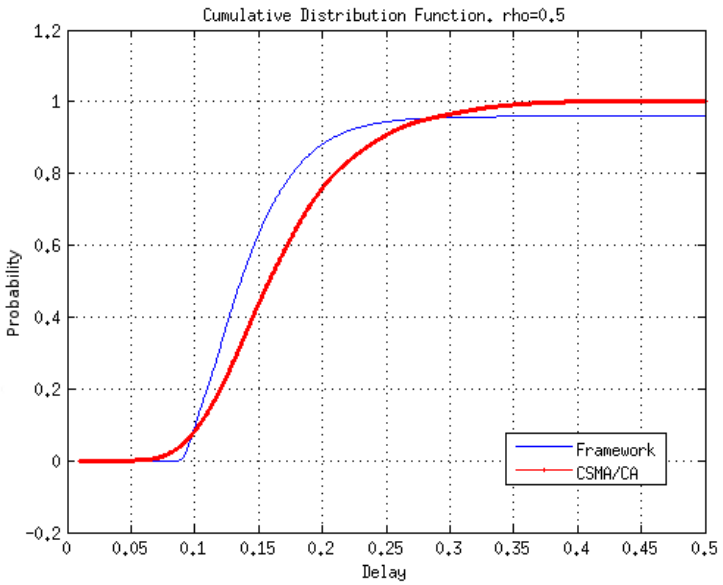


Figure 4.4: Comparison between Framework and CSMA/CA with $\rho = 0.5$.

Discussion

Previously, we have presented two scenarios and we have shown the results by comparing the CDF of both theoretical and empirical curves. For the Jackson scenario implemented in JMT, Figure 4.2 shows that both curves fit perfectly so empirical and theoretical results are the same for this queueing network scenario. For the Prowler scenario, we have taken some considerations in order to simplify the model. In particular, we have considered a slightly different approach for determining the delay introduced by the CSMA/CA since we do not have probability p that

a node finds the medium idle when it attemps to transmit. An approach for solving this can be found in [Zhou and Mitchell, 2010]. In this way, we have done a simplification in our model when analysing the delay introduced by CSMA/CA. As we mentioned before, the probabilistic QoS only requires that the e2e delay should not exceed a given threshold with certain probability. So this approximation still gives significant result. We can see from Figure 4.3 and 4.4 that, for these two cases with low and medium traffic load, for a given threshold, say $250ms$, the theoretical curves provide satisfying e2e delay meet probability $P[e2e\ delay > 250ms]$. Table 4.3 summarize these results.

In the case of $\rho = 0.1$ where the load traffic is low and the channel is almost all the time

Scenario ($\rho = 0.1$)	$P[e2e\ delay \leq 250ms]$
Framework	0
CSMA/CA	0
Scenario ($\rho = 0.5$)	**$P[e2e\ delay \leq 250ms]$**
CSMA/CA	0.093
Framework	0.056

Table 4.3: e2e delay meet probability.

idle, the extra delay due to the backoff process can be considered almost negligible and thus, the model we have done for estimating the delay in node is not so far from the empirical one as shown in Figure 4.3. For the case of $\rho = 0.5$, an increase in the number of times a message enters in backoff is expected since the probability of finding the channel idle is lower than the case of $\rho = 0.1$. This can be seen in Figure 4.4 where both curves are not so close to each other. However, results are still acceptables taking into account the e2e delay meet probability.

4.3 TKN154 Module

In this section, we give an overview of the TKN154 implementation of the IEEE 802.15.4 MAC protocol over TinyOS since it was used to configure the testbed scenarios. TKN154 [Hauer, 2009] is a platform independent IEEE 802.15.4-2006 MAC implementation for the 2.1 release of the TinyOS execution environment meeting the main tasks of the 802.15.4 MAC protocol such as PAN association and disassociation, slotted and unslotted versions of protocol, beacon transmission and synchronization, among others. Main components and interfaces used to exchange MAC frames between components are shown in Figure 4.5 and defined in [Hauer, 2009]. TKN154 MAC can be divided into three sublayers. The lowest level, the RadioControlP component, manages the access to the radio. Components on the second level represent different parts of a superframe. For instance, BeaconTransmitP/BeaconSynchronizeP responsibles for transmission/reception of a beacon frame, DispatchSlottedCsmaP component manages frame transmission and reception during the CAP. The components on the top level implement the remaining MAC data and management services such as PAN association or requesting data from a coordinator. A component of this level typically provides MAC primitives to the next higher layer. For instance, DataP provides *MCPS-DATA.request* primitive to the next higher layer to send a frame to a peer device. Data frame will be assembled and enqueued in the send queue DispatchQueueP. DispatchSlottedCsmaP will eventually dequeue the frame and manage its transmission. Transmission status will be propagated to higher layer by means of *MCPS-DATA.confirm* event where an appropriate status code will signal whether the transmission was successful or not.

A set of interfaces towards the Radio Driver are also implemented. These interfaces push many time-critical operation from the MAC to the radio driver which are not negligible affecting the packet transmission delay.

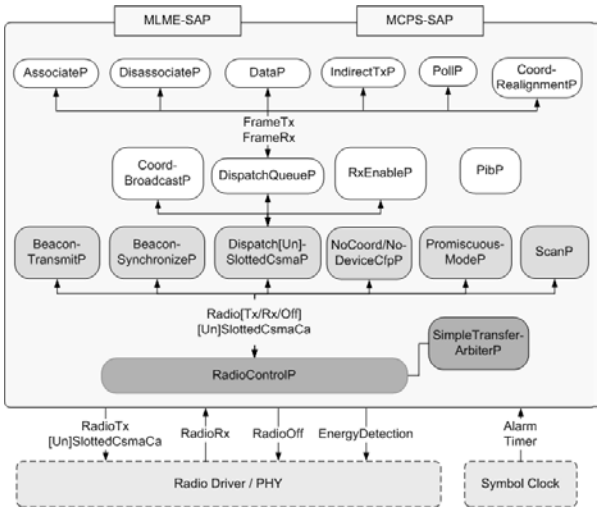


Figure 4.5: The TKN15.4 architecture: components are represented by rounded boxes, interfaces by connection lines.

4.4 Impact of Duty Cycle Mechanism on the Packet Delay in IEEE 802.15.4

One of the most importants issues in WSN concerns the energy optimization. This fact is important in order to increase the network lifetime as well as to avoid having disconnected nodes that can complicate the well-functioning of the whole network. As seen in section 2.3, one of the strategies for increasing the node's lifetime is known as *duty cycle*. The idea behind the duty cycle is to allow nodes to alternate between the states *sleep* and *asleep* saving the energy when nodes turn off the radio and go to sleep mode. Naturally, this mechanism introduces some non-negliglible problems. The fact of having a set of nodes in sleep mode may introduce some constraints in terms of packets delay, for instance, when packets arrive to a node in sleep mode. Our objective is to analyse the impact of the duty cycle over packet's delay within a node running the standard IEEE 802.15.4 MAC protocol in a real WSN scenario. In [Huang et al., 2009], authors analyse IEEE 802.15.4 duty cycle operation. Performance metrics, including queuing drop rate, goodput, and power consumption, are evaluated using a proposed analytical model and NS-2 based simulation. Authors in [Neugebauer et al., 2005], proposed a new algorithm for BO adaptation in IEEE 802.15.4 and analyzed the performance using their own developed simulator. Shu et al. [Shu et al., 2006] implemented a C-based simulation model for IEEE 802.15.4 MAC to determine optimal BO and SO values such that overall network energy consumption is minimized. However, results in all of these works were validated by simulation and not by considering a real environment with real motes. In this section we present an analysis of the IEEE 802.15.4 duty cycle when considering a real scenario over TinyOS and TelosB motes instead of using a simulation approach. In concrete, we analyse how duty cycle impacts in the delay for successful packet sent and packet dropped rate when considering different configurations of (BO, SO) pairs,

different packet arrival rates, queue size and duty cycle configurations. We consider the delay from the moment a packet arrives to the MAC queue until the reception of the corresponding acknowledgement. Results are presented in terms of the delay and packet drop rate.

4.4.1 Duty Cycle in IEEE 802.15.4

As mentioned in section 2.4.1, the superframe structure in the standard IEEE 802.15.4 defines an active and an inactive period. The length of these periods is defined by two parameters: *macBeaconOrder(BO)* and *macSuperframeOrder(SO)*. The former determines the interval at which the coordinator must transmit beacon frames. The second parameter describes the length of the active portion of the superframe. The duty cycle is the ratio of the length of an active period SD to the length of a BI, and is calculated as $\left(\frac{1}{2}\right)^{BO-SO}$. In this way, by handling both SO and BO we can get different duty cycle configuration. For instance, by setting $BO = 5$ and $SO = 4$ we get a duty cycle of 50% having active and inactive periods of $0.245s$ and a beacon interval of $0.491s$ while by setting $BO = 5$ and $SO = 3$ we get a 25% duty cycle where active and inactive periods are $0.122s$ and $0.369s$ respectively. Additionally, divers length of active and inactive periods can be obtained for a given duty cycle. For instance, by setting $BO = 1$ and $SO = 0$ we also obtain a 50% duty cycle but having short active and inactive periods of $0.015s$. Different pairs under the same duty cycle have differing impacts on throughput, delay and energy efficiency.

4.4.2 Experiments

Our objective in this section is to give a comprehensive and experimental investigation on how the delay for succesful packet sent is affected when considering some parameters such as packet arrival rate, buffer size and duty cycle configuration. Specially, we study the impact in the delay and packet drop rate (which gives an idea on how the throughput is affected) when varying both BO and SO for a given duty cycle. As we explained before, different configurations of BO and SO can be set to obtain a specific duty cycle. We explore then the best configuration of BO and SO in terms of delay and packet drop rate for different input parameters. We consider a star topology with a centralized coordinator and four devices or sensor nodes are located around the coordinator, as depicted in Figure 4.6. Distance between devices and coordinator is the same for all scenarios and was set to 1 meter. The transmission power for each node was set to 0dBm. We know from [Zuniga and Krishnamachari, 2004] that, for this transmission power and considering a distance of 1 meter, the packet reception rate is almost 1 since the transitional region (a region characterized by unreliable and asymetric links with high variance in reception rate) starts at a distance of almost 10 meters. Packet payload is fixed as 34 bytes for all scenarios. Channel bitrate is 250kbps. Default values of MAC parameters are summarized in Table 4.8. We considered two different values of duty cycle: 25%, 50%. (BO, SO) pairs for each duty cycle $\mathcal{DC}$ are specified as follows:

$$\begin{aligned} \mathcal{DC}_{25} &= \{(4,2),(5,3),(6,4),(7,5),(8,6),(9,7)\} \\ \mathcal{DC}_{50} &= \{(3,2),(4,3),(5,4),(6,5),(7,6),(8,7)\} \end{aligned} \tag{4.8}$$

The arrival rate λ is taken from the set $\mathcal{A} = \{1, 10, 20, 50\}$ packets per second and buffer length K on each node from $\mathcal{K} = \{5, 20\}$ packets. The experimentations were done as follows. For a given value of λ, buffer length K and duty cycle $\mathcal{DC}_{\#}$ we took the first pair of (BO, SO) for the current duty cycle and we measured the delay d_i and packet drop rate r_i for the current execution i. The duration of the execution is 3 minutes.We repeat the same experiment 10 times and we

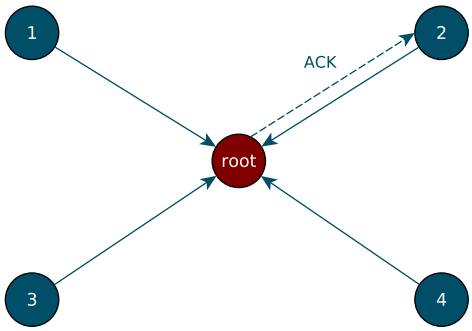


Figure 4.6: Star Topology Scenarios.

compute the average delay $\tilde{d}_{BO,SO} = \frac{1}{10}\sum_i d_i$ and average packet drop rate $\tilde{r}_{BO,SO} = \frac{1}{10}\sum_i r_i$ for the current pair (BO, SO). We then repeat the whole process keeping both λ and K but moving to the next pair (BO, SO) for the current duty cycle. This process is repeated until we have finished with all duty cycle pairs of $\mathcal{DC}_{\#}$. We then plot the average delays and packet drop rate for each (BO, SO) found previously. The same process is repeated for all λ values keeping the same duty cycle $\mathcal{DC}_{\#}$ and buffer length K. Once we have finished we proceed to change the buffer length K and we repeat exactly the same procedure for this new buffer length value, always considering the same duty cycle $\mathcal{DC}_{\#}$. Finally, we repeat the whole process for the other duty cycles $\mathcal{DC}_{\#}$. Table 4.9 shows each of the scenarios.

Parameter	Value
Max Frame Retries	3
Max CSMA Backoff	4
Max Backoff Exponent	5
Min Backoff Exponent	3
Battery Life Extension	False

Table 4.4: MAC parameters on each node

Duty Cycle	λ (packets/second)	BO	SO	Buffer Length
25%	1, 10, 20, 50	4	2	5, 20
	1, 10, 20, 50	5	3	5, 20
	1, 10, 20, 50	6	4	5, 20
	1, 10, 20, 50	7	5	5, 20
	1, 10, 20, 50	8	6	5, 20
	1, 10, 20, 50	9	7	5, 20
50%	1, 10, 20, 50	3	2	5, 20
	1, 10, 20, 50	4	3	5, 20
	1, 10, 20, 50	5	4	5, 20
	1, 10, 20, 50	6	5	5, 20
	1, 10, 20, 50	7	6	5, 20
	1, 10, 20, 50	8	7	5, 20

Table 4.5: Scenarios

4.4.3 Results

In this section we present the results we have obtained from the experimentations. We present results in terms of the average delay from the moment a packet arrives to the MAC buffer until the acknowledgement is received. We also present results of the packet drop rate for each of the scenarios as well as the percentage of deferred transmissions. Figures 4.7, 4.8, 4.9 and 4.10 summarize results for the packet delay and the packet drop rate for each of the scenarios. X-axis matches the ordered set of pairs (BO, SO) for the corresponding duty cycle $\mathcal{DC}$ as shown in (4.8). Finally, Tables 4.6 and 4.7 show the percentages of deferred transmissions for each scenario.

$\mathcal{DC}$	λ	Buffer Size = 5					
		$(4,2)$	$(5,3)$	$(6,4)$	$(7,5)$	$(8,6)$	$(9,7)$
25%	1	3.8	1.5	0.7	0.3	0.14	0.08
	10	12.4	6.9	2.6	1.12	0.4	0.18
	20	15.3	7	3.4	1.5	0.8	0.32
	50	12.4	8.1	3.61	1.78	0.9	0.09
		Buffer Size = 20					
	1	3.65	1.18	0.52	0.37	0.096	0
	10	12.3	6.65	2.9	1.7	0.8	0.07
	20	13.4	7.4	4	1.7	0.8	0.5
	50	13.4	7.8	2.8	1.7	0.6	0.3

Table 4.6: % Deferred Transmissions for each (BO, SO). Duty Cycle = 25%

$\mathcal{DC}$	λ	Buffer Size = 5					
		$(3,2)$	$(4,3)$	$(5,4)$	$(6,5)$	$(7,6)$	$(8,7)$
50%	1	5.7	2.8	1.6	0.8	0.38	0.18
	10	13.1	6.3	2.4	1.2	0.6	0.26
	20	12.4	7.1	3.6	1.4	0.72	0.38
	50	12.9	7.7	3.6	1.5	0.85	0.47
		Buffer Size = 20					
	1	4.7	2.4	1.5	0.8	0.24	0.13
	10	11.4	6.3	3.27	1.79	0.66	0.25
	20	13.3	7.4	3.79	1.42	0.81	0.27
	50	13.3	8.82	3.83	1.4	0.72	0.3

Table 4.7: % Deferred Transmissions for each (BO, SO). Duty Cycle = 50%

4.4.4 Discussion

We start by analysing those scenarios with low traffic load where the arrival rate $\lambda = 1$. Based on the packet drop rate we can characterize three types of scenarios: low traffic for those cases where the packet drop rate is zero and moderate and saturated traffic scenarios for those cases where packet drop rate is around 50% and above 80%, respectively. As we can see from Figures 4.7, 4.8, 4.9 and 4.10, when $\lambda = 1$ the packet drop rate is almost zero. Besides, arrival rate is very low for this scenario and thus, buffer in nodes is empty almost all the time. In these cases, delay for low traffic scenario increase exponentially when increasing BO and SO. The reason is that, even that we keep the same duty cycle, increasing BO and SO gives a new superframe configuration where the inactive period is extended. That means that the packet delay will increase for those arriving in the inactive period. Conversely, when BO and SO decrease, delay will decrease as well but in this case the number of beacon packets exchanged will increase and then the overall energy consumption. However, since buffer is almost empty and packet drop rate is almost zero, the throughput is not affected in this case. Hence, for low traffic scenarios it should be necessary to find a trade-off between energy consumption and packet delay.

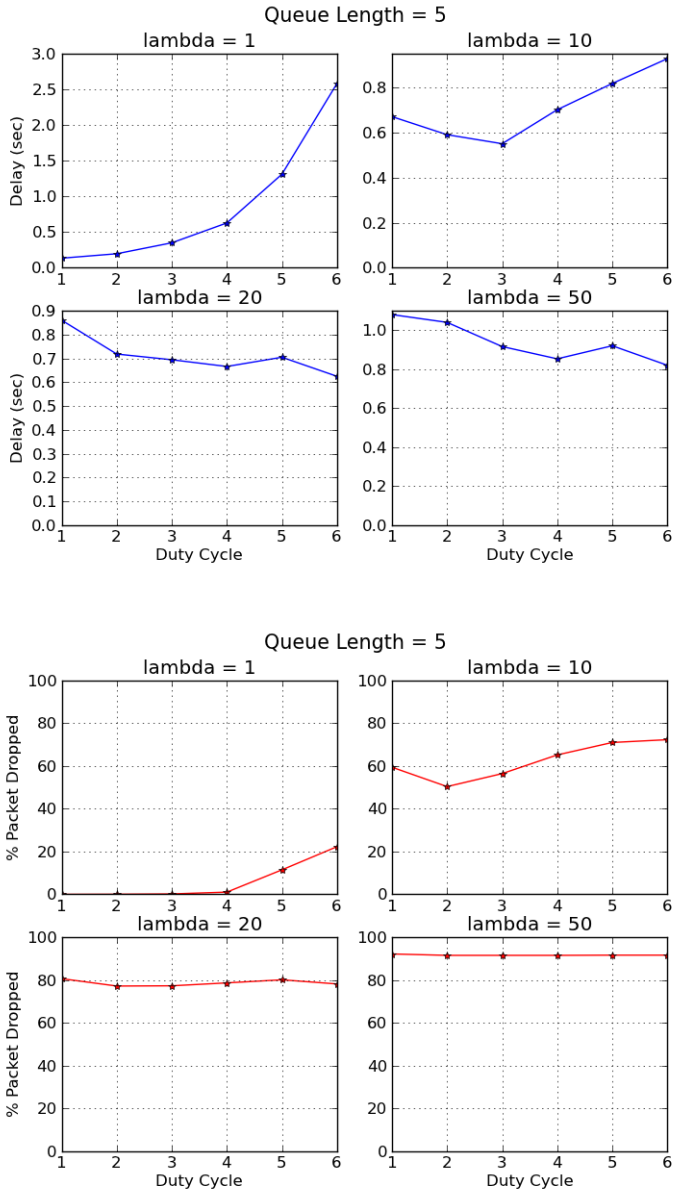


Figure 4.7: Duty Cycle = 25%, Buffer Length = 5.

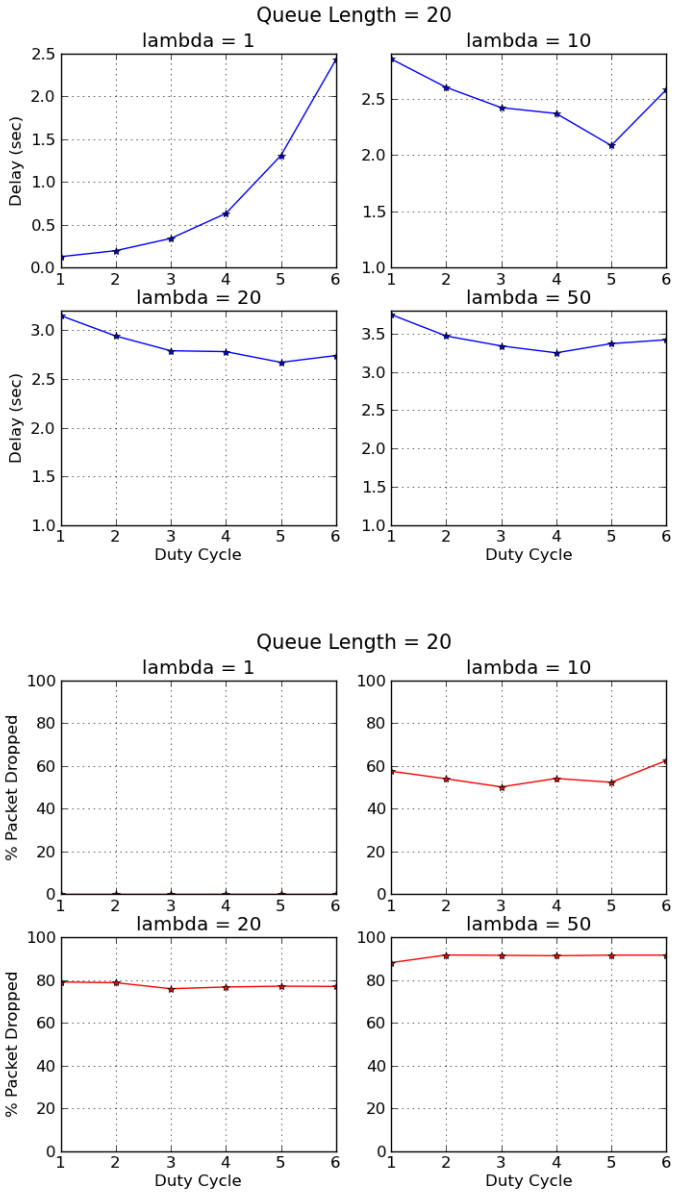


Figure 4.8: Duty Cycle = 25%, Buffer Length = 20.

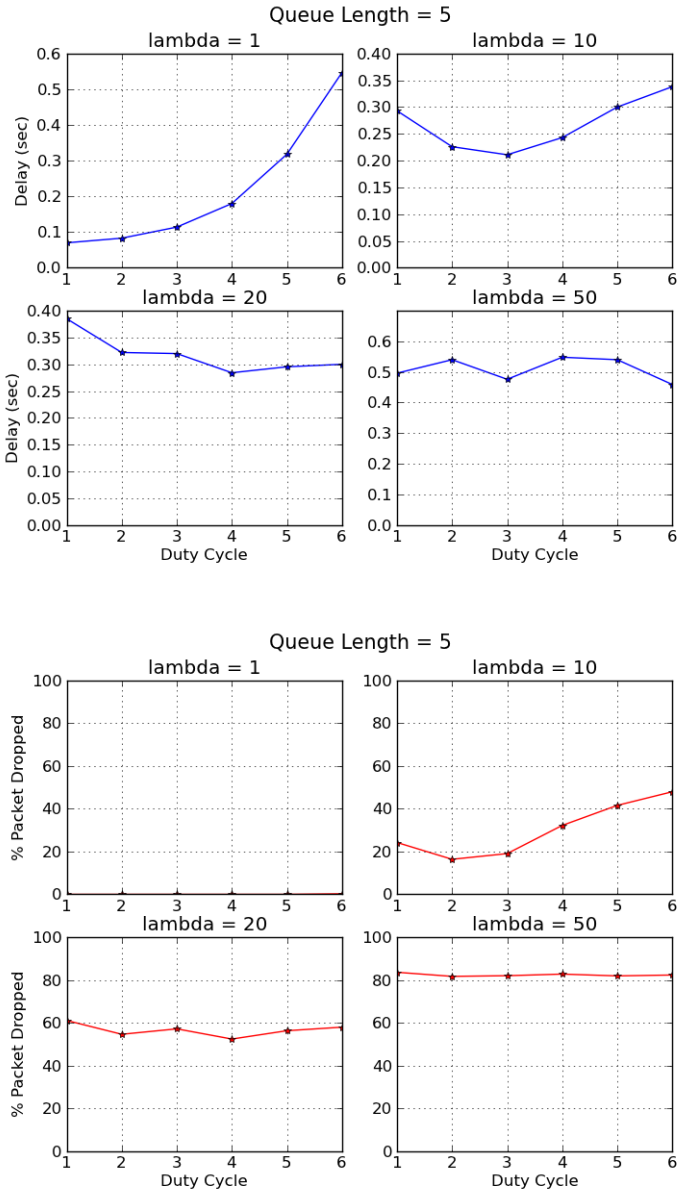


Figure 4.9: Duty Cycle = 50%, Buffer Length = 5.

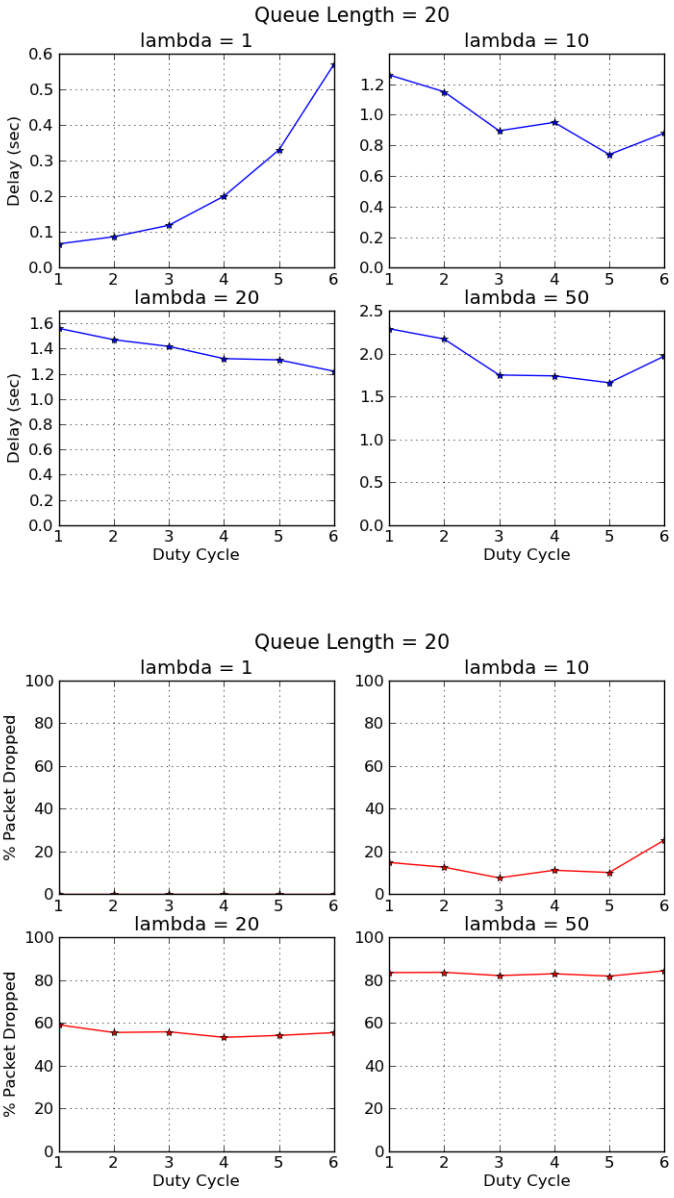


Figure 4.10: Duty Cycle = 50%, Buffer Length = 20.

A more interesting result is obtained for almost saturated scenarios. When considering a 25% duty cycle and both $\lambda = 20$ and $\lambda = 50$, as well as for the case of 50% duty cycle with arrival rate $\lambda = 50$, we observe that packet drop rate is above 80% meaning that node buffer is full almost all the time. Looking at the corresponding delay graphics we can see that in almost all cases delay starts by decreasing while incrementing BO and SO until reaching certain points. After this threshold, the delay, either increases, as seen for both duty cycle boundaries configuration and *buffer size=20* or fluctuates between upper and lower boundaries not so far from each other, as seen for both duty cycles and *buffer size=5*. The only exception, where delay do not decrease but starts fluctuating from the beginning, is the case of 50% duty cycle and *buffer length=5*. In order to understand this behavior we have to take into consideration another point that we have not mentioned before and is related to the number of transmissions deferred. On one hand, as we explained in section 4.4.1, transmission is started if and only if the remaining number of backoff periods in the current superframe is enough to handle both the frame and the subsequent acknowledgement transmissions. Otherwise, transmission is deferred until the next superframe. On the other hand, as (BO, SO) increases, the length of superframe also increases and then both active and inactive periods (always keeping the same $\mathcal{DC}$). When looking at the percentage of transmissions deferred for each of the (BO, SO) configuration, we can see that the behavior is always the same independently of the scenario: percentage of deferred transmissions for $(BO+1, SO+1)$ is almost half the percentage of (BO, SO). This is reasonable since a short superframe results in short active period and then the percentage of deferred transmission for short superframe is greater than the case of a long superframe. This issue will have an impact in the delay of those packets waiting in the buffer, specially in saturated scenarios where buffer is full almost all the time. As we increase the superframe length, we increment the length of the active period reducing the percentage of deferred transmission and then the average packet delay for those enqueued packets. However, increasing the superframe also increase both active and particularly the inactive period. After certain point, the length of the inactive period starts to impact in the packet delay, as seen in most of the cases, incrementing the delay for those packets waiting in buffer as well as those packets arriving in the inactive period. In summary, delay is affected first due to a short length of the active period and percentage of deferred transmission and then, as we increment (BO, SO), by the length of the inactive period. This behavior is not seen in low traffic scenarios since the buffer for these cases is empty almost all the time. Then, delay is affected only by the length of the inactive period and not by the deferred transmissions. Authors in [Huang et al., 2009] have shown that a large BO increase the packet drop rate due to an extended inactive period. We can also see this behavior, specially for those scenarios with low traffic load as seen in Figure 4.7 for both $\lambda = 1$ and 10, as well as in Figures 4.8, 4.9 and 4.10 for $\lambda = 10$. On the other hand, for saturated scenarios the packet drop rate is not affected by the length of the inactive period but for the fact that buffer is full almost all the time.

As explained before, when both BO and SO decrease, the energy consumption increase due to the increment in the number of beacon packets exchanged. Additionally, energy consumption in saturated scenarios is also affected due to an increment in the channel contention activity at the start of the next superframe, since node's buffer is almost always full. In these scenarios, the number of drop packets remains almost constant so there is no significant impact in the throughput, that is to say, throughput is poor independently of the (BO, SO) configuration. For saturated scenarios, since packet drop rate remains the same for all configurations, selecting the (BO, SO) configuration that minimises the average delay seems to be the best option achieving also a reasonable energy consumption.

The same reasoning to explain why delay decrase to certain point and then increments or fluctuates can be applied for the cases of moderate traffic as seen in scenarios of 25% duty cycle,

buffer length=5 and $\lambda = 10$ as well as 50% duty cycle and $\lambda = 20$ for both buffer size. For those scenarios packet drop rate is around 60%. On the other hand, for moderate traffic scenarios, throughput is improved respected to saturated scenarios due to a lower rate of drop packets. Energy consumption is expected to be lower when comparing with saturated scenarios since the channel contention activity at the start of the next superframe decreases. Finally, a couple of results concerning those scenarios between low and moderate traffic as seen in 50% duty cycle and arrival rate $\lambda = 10$. As we can see, delay behaves in the same way as saturated and moderate scenarios. However, packet drop rate starts decrasing until reaching the (BO, SO) configuration that minimises the delay after which both packet drop rate and delay starts increasing. Selecting this configuration will give us an optimal solution in terms of both delay and packet drop rate keeping a moderate energy consumption. Finally, a question that may arise when looking at the high percentages of packets drop. Why not considering a large buffer length in order to avoid having high rates of packets drop ?. The answer is that we are working on TelosB motes which works with not enough memory so defining a large buffer would simply not be feasible.

4.5 Comparision Between Misic's Model and Measurement Approach

As previously discussed, a lot of research work has been done to model the network through different methods, including analytical modelling and simulation based analysis. Due to the stochastic nature of WSNs, analytical models are based on Markovian approaches whose objective is to model the stochastic behavior of the underlying MAC protocols in WSNs. Misic et al. in [Misic and Misic, 2008] proposed a quite complete Markov chain model for modelling the behavoir of the slotted version of the IEEE 802.15.4 MAC protocol. Here, authors analyse the behavoir of the slotted version of the standard MAC protocol considering active and inactive periods, deferred packets as well as a simple physical model. Each node is modeled as a M/G/1/K queueing system.
They consider a single 802.15.4 cluster with star topology that operates in beacon enabled, slotted CSMA-CA mode, with uplink traffic only. The cluster consists of n devices and packet arriving to each device follow a Poisson process with the mean arrival rate of λ and each node accepts new packets through a buffer with the finite size of K packets. The operation of the cluster is modeled using the theory of discrete time Markov chains and M/G/1/K queues. It is important to note that the system is modeled as a discrete time Markov chain since time is discretized in backoff periods. Based on the protocol specification [802, 2007], authors model the system as a stochastic process as detailed in section 3.3.2. Resulting Markov chain is shown in section 3.3.2. From the stationary distribution of the Markov chain, authors were able to find the probability distribution of the packet service time, probability distribution of the queue length and both probability distribution and average delay from the moment a packet arrives to the queue until the moment the node receives the corresponding acknowledgement. In this section we present an experimental evaluation of the Slotted CSMA/CA MAC protocol by considering realistic conditions using TelosB motes and an implementation of the protocol over TinyOS. Results are then compared with the ones that we obtained by applying the Misic's model presented in [Misic and Misic, 2008]. We present results in terms of average delay showing the gap between theoretical and empirical approaches and we give some implementation considerations that needs to be taken into account when designing theoretical models for evaluating the delay in WSN in order to have a more accurate model to work with.

4.5.1 Delay Analysis in TKN154 Implementation

As we mentioned before, and based on simulation results, Misic's model seems to be accurate for estimating the delay within the slotted IEEE 802.15.4 MAC protocol. However, implementation considerations were not taking into account and, as we will see next, these cannot be omitted if we want to have a suitable mathematical model for estimating the delay. In this subsection we analyse the TKN154 components interaction showing how delay is affected due to the implementation issues.
In TinyOS there are two threads of execution: tasks and hardware event handlers. Tasks are functions whose execution is deferred. Once scheduled, they run to completion and do not preempt one another. Hardware event handlers are executed in response to a hardware interrupt and also runs to completion, but may preempt the execution of a task or other hardware event handler. In this way, the completion of a particular task or event handler may be delayed due to a hardware interrupt affecting the delay of the whole protocol. Some other issues associated to the Operating System behavior may have effects on the protocol performance. For example, the TinyOS interface to the SPI bus introduces a large processing overhead that reduces the achievable SPI bus transfer rates [Suriyachai et al., 2009].

4.5.2 Extra-delays estimation

In order to estimate extra-delays in the implementation of the MAC protocol over TinyOS we have analysed the execution stack from the moment a packet is generated until the moment the packet is acknowledged (or eventually lost if the number of retries exceeds the predefined threshold). Figure 4.11 shows a sequence diagram with the most relevant operations within the protocol. The first non-negligible delay presented in the execution is related to the SPI resource request in step 6. This delay together with the CC2420 switch to Rxmode shown in step 7 gives an extra delay of $t_1 = 5ms$. Then a random backoff period and two clear channel assessment (CCA) follows the execution of the protocol. However, both CCA and the random backoff period were taken into account in [Misic and Misic, 2008]. Next, a fix delay (step 11) of $t_2 = 2.8ms$ between the end of the second CCA and the invocation of the *transmissionStarted* function was found. Transmission is then delayed $t_3 = 2.8ms$ due to SFD Capture operation (step 13). Finally the packet is sent with a transmission delay shown in step 14. We have also found an extra-delay (step 15) $t_4 = 3ms$ in the coordinator side before sending the acknowledged packet back to the source (step 16). This analysis gives us un idea of the extra-delay due to the protocol implementation that should be taken into account when estimating the MAC protocol delay in TinyOS. Finally, we have also found a random delay t_5 due to deferred packets. Normally, when the remaining time within the CAP period of the current superframe does not suffice to complete the transmission the packet is deferred and will be transmitted at the begining of the next superframe. However, in TinyOS we have seen that in some cases the deferred packet transmission does not start at the begining of the next superframe. Instead, it skips the immediate superframe deferring the transmission to the next one. That means that in this case, packet transmission will be delayed $t_{wait} = t_{curr} + t_{sup}$ where t_{curr} is the remaining time within the CAP period of the current superframe and t_{sup} is the duration of the superframe (in our case a 100% duty cycle is consider so optional inactive period is not taken into account). Considering that in our experimental scenarios, MAC attributes SO and BO were set to five then $t_{sup} = 30720$ symbols meaning that delay in those packets skiping the immediate superframe would be at least 30720 symbols = 0.5 seconds (1 symbol = 16μs).

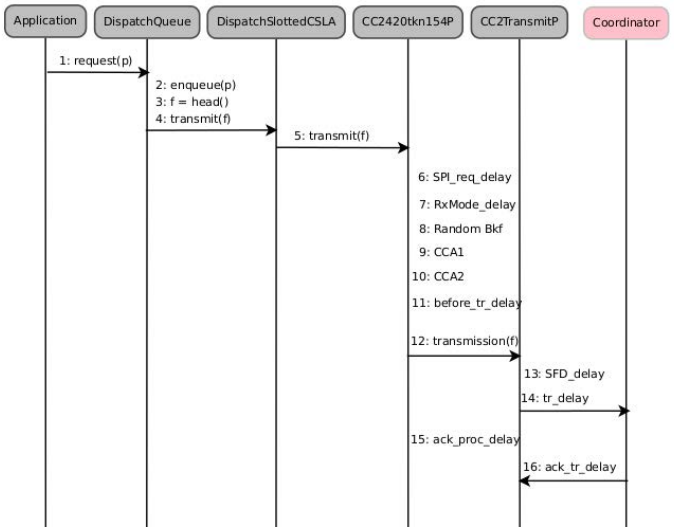


Figure 4.11: TKN154 Delay Analysis.

4.5.3 Experiments

In this section we present the experiments we have done in order to compare the theoretical results obtained by [Misic and Misic, 2008] with real scenarios using an implementation of the slotted IEEE 802.15.4 MAC protocol over TinyOS and TelosB motes. We first start by specifying the main parameters and then we present a set of scenarios we have used in the experimentation.

Parameters & Scenarios

In order to compare both theoretical and experimental results we have set the same MAC protocol parameters which are shown in Table 4.8. As we can see both BO and SO are set to 5 so no

Parameter	Value
Max Frame Retries	3
Max CSMA Backoff	4
Max Backoff Exponent	5
Min Backoff Exponent	3
Battery Life Extension	False
Beacon Order	5
Superframe Order	5

Table 4.8: MAC parameters.

inactive period is considered and then Duty Cycle is 100%. We consider six scenarios which varies in the number of nodes and arrival rate of packets to nodes. The idea is to make a comparition of both approaches in terms of the average delay, that is to say, the average delay from the moment a generated packet is put in the queue until the reception of the acknowledged for that packet. We consider a star topology where coordinator is situated at the center and devices are located around the coordinator. Distance between devices and coordinator is the same for all scenarios and was set to one meter. The transmission power for each node was set to 0dBm. We know from [Zuniga and Krishnamachari, 2004] that, for this transmission power and considering

a distance of one meter, the packet reception rate is almost 1 since the transitional region (a region characterized by unreliable and asymetric links with high variance in reception rate) starts at a distance of almost 10 meters. Simulation parameters are summarized in Table 4.9. Packet payload is fixed as 34 bytes for all scenarios. As in [Misic and Misic, 2008], packet arriving to each device follow a Poisson process with mean arrival rate of λ and each node accepts new packets through a buffer with finite size of $K = 2$ packets. All scenarios use the same buffer size. Channel bitrate is 250kbps.

Scenario	Nodes	Traffic Load λ (packets/s)
Scenario 1	2	1
Scenario 2	4	1
Scenario 3	6	1
Scenario 4	2	10
Scenario 5	4	10
Scenario 6	6	10

Table 4.9: Scenario Parameters

Results

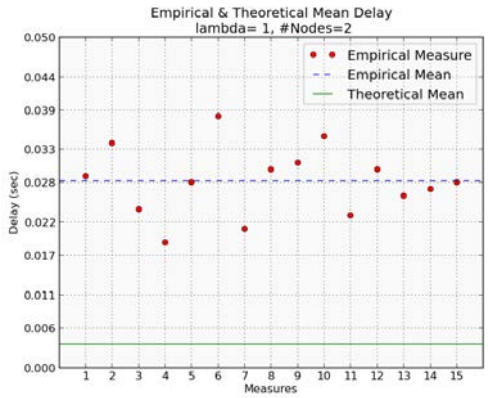
We present now the experimentation results. Our objective is to compare the theoretical and empirical average delay. Misic framework was implemented in Matlab while we use TKN154 implementation of the slotted IEEE 802.15.4 over TinyOS and TelosB motes. For each scenario, a total of fifteen measurements were done in TelosB motes and then the average delay was found and compared with the average delay obtained by the [Misic and Misic, 2008] framework. Table 4.10 shows theoretical and empirical average delay for each scenario. Figures 4.12 and 4.13 summarize the results of each scenario. Points in graphics represent the empirical measures (a total of fifteen instances of measurement for each scenario). Each empirical measure (point in graphics) is the result of the analysis of all the packets that were generated during the current instance of measurement. We took the average of these delays in order to determine the empirical measure of delay for the current instance and we plot the corresponding point. This procedure is repeated fifteen times for each scenario. Blue-dashed line represents the empirical average delay (average of fifteen points) for each scenario, while the green line shows the average delay obtained by [Misic and Misic, 2008] mathematical framework.

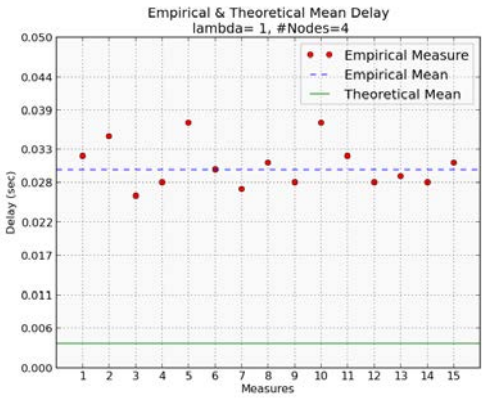
Scenario	Theoretical Av. Delay (sec)	Empirical Av. Delay (sec)
Scenario 1	0.00356	0.028
Scenario 2	0.0036	0.030
Scenario 3	0.0036	0.031
Scenario 4	0.0038	0.040
Scenario 5	0.0043	0.042
Scenario 6	0.005	0.044

Table 4.10: Theoretical VS Empirical Delay.

4.5.4 Discussion

As we can see from Figure 4.12, 4.13 and Table 4.10 it is evident that there is an important gap between empirical and theoretical results. We see from results that when the number of





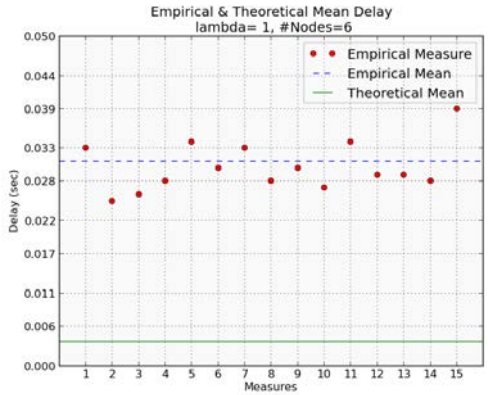


Figure 4.12: Traffic load $\lambda = 1$ packets per second

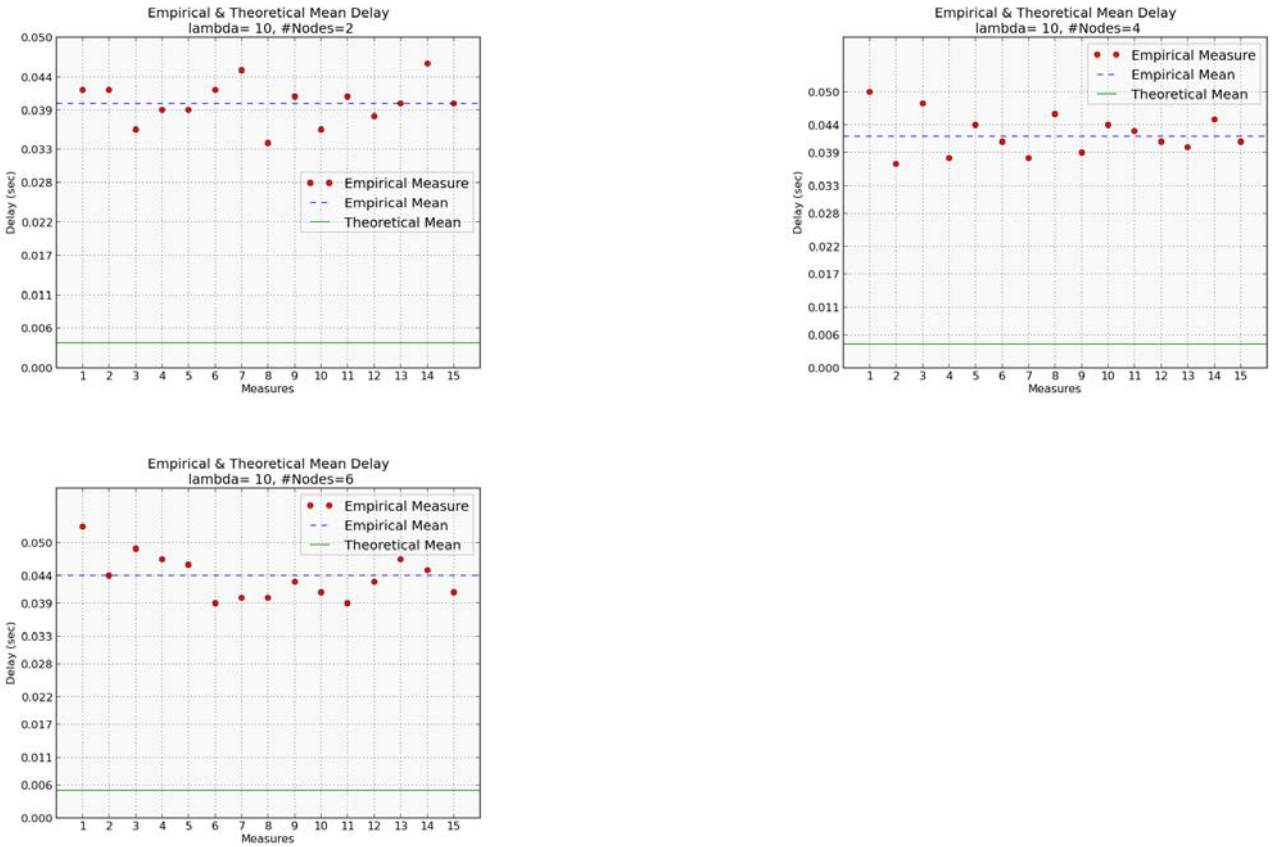


Figure 4.13: Traffic load $\lambda = 10$ packets per second

nodes increases the delay also increases. This is normal since as the number of nodes grows, the number of times the channel is found busy also increments. This behavior is expected as well for the number of collisions. We also see that delay when considering a mean arrival rate of $\lambda = 10$ is greater than the one found for scenarios having $\lambda = 1$. This is due to the fact that for scenarios having mean arrival rate of $\lambda = 10$, the queue is expected to be busy during a non negligible time. That means that delay for those packets in the queue will increase since it would have to wait until the acknowledgement from the previous sent packet arrives. An important issue we found is regarding the first scenario. In this case, we have two nodes and mean arrival rate $\lambda = 1$ so we do not expect to have too much collisions, channel is expected to be idle most of the time and, in theory, the queue should be in idle state most of the time. However, the gap between theoretical and empirical delays is almost 24ms. How can we explain this behavior ?. By analysing the TinyOS traces during the experiment we found that a number of deferred packets skips the immediate superframe delaying the transmission to the next one. For these cases, the node's queue would not be necessarily idle most of the time as, in theory, we would expect for this low-traffic scenario. This happens because deferred packets would have an extra-delay and then arriving packets would find the node busy and they will wait until the deferred packet is acknowledged. Then, contrarily to the expected behavior, the queue won't be idle most of the time and this fact will impact in the average packet delay. Markov chain model in [Misic and Misic, 2008] is a very good approach for modelling the IEEE 802.15.4 MAC protocol considering and covering the main aspects of the protocol behavior (packet collisions, packet deferring, random backoffs, etc). Simulation results shown in [Misic and Misic, 2008] validate the mathematical approach. However, as we discussed in section 4.5.2, non negligible delays arise when consider a realistic scenario with real motes due to operations of the underlying operating system. From the practical point of view, omitting the operating system features and behavior would leads to a useless model for analysing the protocol performance, in this case, the delay. A question that emerges immediately: is it possible to reduce the gap between theoretical and empirical results?. The first thing that we can do in order to reduce the gap is to focus our attention in the protocol implementation. In our analysis in section 4.5.2, we detected an anomaly in the TKN154 implementation concerning the deferred packets. As we saw, some deferred packets skip the incoming superframe delaying the transmission until the reception of the next one (which amounts 0.5 seconds for the set of parameters defined here). So an important point to do is to detect anomalies in the implementation that could introduce extra-delays in the protocol and fix them. However, as shown in Figure 4.11, even considering an implementation exempt from anomalies and errors, extra delays are present due to function calls, hardware operations such as resources request, radio switch to reception/transmission mode, etc. Considering an error-free implementation, one approach then to reduce the gap is to quantify the existing delays in the implementation as done in section 4.5.2 and add them to the theoretical model in such a way that extra-delays are also considered when computing the average delay, for instance by changing formule (3.36) in [Misic and Misic, 2008].

4.6 Conclusion

In this chapter we have presented some preliminary that we have obtained regarding the delay performance on WSNs. We first gives an empirical support and evaluation of a theoretical framework based on frequency domain for estimating the end to end delay in a multi-hop network. Our study shows that the proposed analytical approach is able to estimate the end to end delay distribution when considering queueing delay in nodes for the case of low and moderate traffic.

However, one of the drawbacks on this approach is the fact that estimating the delay distribution on each component is normally not straightforward. In our example, we consider a Jackson Network and therefore, service time on each node is supposed to be exponential. Nevertheless, this assumption cannot be applied in general WSN scenarios. Further in this thesis, we will show how we make use of this model for estimating the end to end delay distribution in a multi-hop transmission scenario. Next, and considering that most important MAC protocols implements the duty cycle mechanism as the strategy for saving energy, we presented results showing how packet's delay as well as the percentage of dropped packets are affected by this mechanism for differents arrival rates and buffer queue length scenarios. For several duty cycles we have studied different (BO, SO) configurations and we have presented results in terms of the average delay from the moment a packet arrives to the MAC buffer until the corresponding acknowledgement is received, as well as the percentage of packets drop rate. For a given duty cycle $\mathcal{DC}_{\#}$, we have analysed several (BO, SO) configurations by changing the traffic arrival rate and node buffer length. We have seen that for scenarios having low traffic, average delay increases exponencially as BO and SO increase. In order to find the best configuration a trade-off between average delay and energy consumption must be taken into account since small (BO, SO) introduces more overhead due to an increment in the beacon exchange. For saturated scenarios, we have shown that there is a (BO, SO) configuration which minimises the delay keeping a reasonable energy consumption. Performance in terms of packet drop rate for these scenarios is poor since buffer node is almost all the time full. We have obtained the same conclusion also for the moderate scenarios except that in this case, both energy consumption and packet drop rate were improved. For scenarios with traffic between low and moderate, an optimal (BO, SO) configuration in terms of delay and packet drop rate was found. We were also able to verify the results obtained in [Huang et al., 2009] with respect to the packet drop rate. We conclude that, for low traffic scenarios, a large BO increase the drop rate due to an extended inactive period.
Finally, we have presented an analysis of the average delay in slotted IEEE 802.15.4 protocol in real scenarios considering the TKN154 implementation over TinyOS. Our objective was to determine the gap between a Markov chain approach for estimating the average delay presented in [Misic and Misic, 2008] and a real implementation in TinyOS. By analysing the execution of the protocol in real nodes, we were able to determine constant and random delays (such as hardware event handlers threw due to hardware interrupts) whose execution can preempt a particular task and thus delaying its completion. Also, we have noticed that some deferred packets skip the immediate superframe, delaying the transmission to the next one. Work presented in [Misic and Misic, 2008] is a well-achieved model which considers the most important aspects of the IEEE 802.15.4 MAC protocol. However, from the obtained results of our work, we can conclude that this theoretical model is not accurate enough since non-negligible issues inherent to the protocol implementation as well as the underlying operating system running on nodes, are not taken into account. Hence, and in order to have an accurate model for estimating the most important parameters, issues inherent to the protocol implementation cannot be ignored.
In the next chapter, we introduce a novel approach for infering a Markov chain model from a MAC protocol execution traces. Our methodology consist in combining measurement and analytical approaches for modelling the MAC protocol behavio. Our idea then, is to take advantage of the measurement approach in order to capture the main features of the WSN execution, specially, those concerning the underlying operating system. In this way, we expect to overcome problems as the one described above (section 4.5).

Chapter 5

A Novel Methodology for End to End Delay Estimation in Wireless Sensor Networks

Contents

5.1 Introduction

In chapter 3, we have presented an overview of the most typical approaches found in literature for estimating the end to end delay in WSNs: simulation models, measurements and analytical models. We have seen that simulation models are normally simplification of the real world and therefore, this approach is not accurate enough for estimating performance parameters. We have also seen that, due to the difficulty to achieve global synchronisation in large-scale sensor networks, the measurement approach for estimating the end to end delay is extremely challenging. Analytical solutions found in literature for modelling the behavior of the MAC layer protocol

for estimating the local delay within a node, were also presented. We have seen that, due to the complexity in mathematical terms for modelling the behavior of MAC protocols in real conditions, most of these analytical approaches simplifies the reality by considering ideal channel conditions, by simplifying the queue model on each node, or even not considering the issues introduced by the underlying operating system. Besides, most of the proposed analytical solutions are conceived for the standard IEEE 802.15.14 MAC protocol and considering a star topology. Hence, the whole delay is estimated in terms of the one-hop delay from a source node to the coordinator located at the center of the topology. The extension of these analytical models for considering a multi-hop topology is not trivial and therefore the delay estimation in a multi-hop transmission scenario is a challenging task.

In this chapter, we propose a novel approach for modelling the network behavior, an approach that combines measurement and analytic approaches overcoming the main problems found in typical approaches. Our approach consists in infering a Markov chain model from network execution traces. In the first section we introduce our approach explaining how we proceed to extract a Markov chain model from execution traces and we give also the approach for estimating the one hop and end to end delays from the extracted model. Then we present three use cases where we show the suitability of our approach to model node's behavior and how this model is used for estimating the end to end delay. The use cases are presented in an incremental fashion. We start by applying our approach considering the slotted version of the IEEE 802.15.4 MAC protocol presenting results for a simple multi-hop transmission scenario. We continue then with a second use case where we apply the approach but changing the underlying MAC protocol. We present then results for a simple multi-hop transmission scenario for the ContikiMAC MAC protocol. These two use cases consider a simple multi-hop scenario having a static routing protocol. In our third use case we make a step beyond by considering also RPL in order to validate our approach when considering also a dynamic routing protocol. We present results in terms of end to end delay for a more complex multi-hop transmission scenario considering X-MAC as the underlying MAC protocol together with RPL routing protocol. Besides, we present a performance comparison in terms of end to end delay distribution and packet reception rate between two RPL routing metrics (hops-count and ETX).

5.2 Methodology

In this section we introduce the design and implementation of our approach which consists in infering a Markov chain model from network execution traces. Given a set of nodes within a wireless sensor network, the objective is to analyse the traces from the execution of each node and to extract a local Markov chain (intrinsic to each node). This Markov chain is obtained based on the execution of the underlying MAC protocol running on each node. To carry out this, a set of steps shown in Figure 5.1 needs to be accomplished. It is important to note that we focus only in the execution of the underlying MAC protocol for extracting the local Markov chain and not, for instance, in the execution of the above routing protocol. This is due to the fact that the MAC protocol runs locally on each node so its execution scope is limited to the node itself while the execution scope of a routing protocol covers the entire network. Hence, and since we are modelling the behavoir of a node, Markov chain states are determined by the specification of the MAC protocol and not by the overlying routing protocol.

In the next subsections we explain each of these steps in order to obtain the mentioned Markov chain and also how this model can be used for estimating the end to end delay in a

WSN.

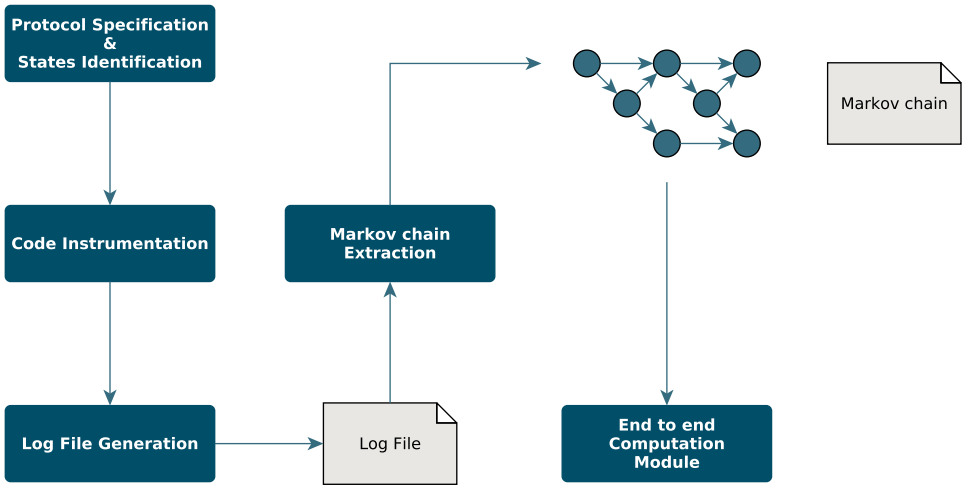


Figure 5.1: Step by step flow diagram of our methodology to estimates end to end delays.

5.2.1 Protocol Specification and States Identification

The first and one of the most important steps in this process is the protocol specification and states identification. Each MAC protocol has a specification from where we can obtain the main aspects of it. Usually, a flow diagram showing the main states and transitions between them is provided. Based on the specification of this protocol we can identify the main states of the protocol (enqueuing packet, sensing, transmitting, packet sent, ...) and the transitions between them. All these states should be taken from the corresponding protocol specification. This step is crucial in order to understand how the MAC protocol works and to suitably accomplish the next step which concerns the code instrumentation. There is another aspect that we should also consider when identifying the states and it is related to protocol implementation. Normally, MAC protocols are implemented based on their specification. Nevertheless, sometimes there are implementation decisions that were not specified in the protocol specification that needs to be taken into account as well. Examples of this were found in the TKN154 [Hauer, 2009] implementation of the IEEE 802.15.4 MAC protocol where some implementation issues that were not specified in the standard were considered. In summary, the state identification should consider both specification and implementation issues in order to suitable model the protocol behavior.

5.2.2 Code Instrumentation & Log File Generation

Previous step is crucial to understand how protocol works. Once we have identified each state and transitions in the protocol, the next step is to generate the log files in such a way that all states and transitions previously identified are present in the protocol execution output (log file). In order to generate this log file, it would be necessary to identify each state in the protocol implementation in such a way that each state will appear in the protocol execution and therefore in the generated log file. The code instrumentation step consists in printing an output line each time a change from one state to another is found in the implementation of the protocol. In this

way, the execution's output will give us a log file showing each state within the protocol together with the corresponding transitions, where transitions are represented by the previous and next state of a given state, that is to say, if state X is followed by state Z and preceded by state Y in the log file, then we have both Y $\rightarrow$ X and X $\rightarrow$ Z in the set of transitions.

5.2.3 Markov chain Extraction & End to End Delay Computation

Markov chain Extraction

The purpose of this module is to detail the extraction process, that is to say, how to extract a Markov chain model from the protocol execution code. The obtained log file gives us the information regarding the states and transitions between states taken from the protocol execution traces. The log file can be grouped into a set of sequences, each one delimited by an initial state and a final state. The initial state is unique and is associated to the instant a packet arrives to the node and is added to the node's buffer. The final states may be more than one, normally associated to the fact of receiving an acknowledge or when a packet is discarded due to a failed transmission. In this way, a log file can be seen as a set of sequence starting at the initial state and finishing in some of the identified final state. Figure 5.2 shows an example of a sequence of states within a log file. Then, the approach consists in processing each of the sequences identifying the

Figure 5.2: An example of a log trace obtained from IOT-LAB Senslab.

states (STATE1, STATE2, STATE3 ...) and transitions between the states within the sequence. A transition from a particular state S is determined by the next state in the sequence. The idea is to determine the number of times (frequency) a transition from one state to another appears in the whole set of sequences. Therefore, after finishing processing the sequences, we will have a set of transition's frequencies from each state to the other ones. This set of frequencies will give us the probability of transition between a particular state to another one allowing also to generate the Markov chain model.

End to End Delay Computation Module

Finally, we implement the end to end computation module (implemented in Phyton) which will compute the end to end delay between a source node n_{src} and the destination node n_{dst}.

In order to estimate this performance parameter we make use of the approach presented in [He et al., 2010b] taking the obtained Markov chain as the input. We first start by computing the delay in one hop for each of the nodes and then we extend the analysis to estimate the end to end delay from source node n_{src} to the destination n_{dst} (sink). In order to estimate the one hop delay we proceed as follows: the Markov chain gives us the information of the transitions between states of the protocol together with the corresponding transition probabilities. Then, it is possible to compute the *Probability Transition Matrix* P. As said before, this Markov chain has a unique initial state s_i and one or more final states $\{s_{f_1}, s_{f_2}, \cdots\}$. Since we are estimating the one-hop delay from the moment a packet arrive to the node until the packet is successfully sent, the only final state we consider for estimating the delay is the one associated to the reception of the acknowledge packet. Without loss of generality, let's call this state s_f. From P and the Laplace transform of the sojourn time[1] distribution e_{s_k} on each state s_k, we are able to compute the *Adjacency Matrix* A defined in [He et al., 2010b] as follows:

$$A = \begin{pmatrix} 0 & p_{s_1 s_2} e_{s_1} & 0 & \cdots & 0 \\ 0 & 0 & p_{s_2 s_3} e_{s_2} & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{5.1}$$

where e_{s_k} represents the Laplace transform of the sojourn time distribution on state s_k and $p_{s_k s_l}$ the transition probability from state s_k to s_l taken from P. To find e_{s_k}, we compute the empirical average sojourn time γ_{s_k} obtained by analysing the traces of the protocol. In order to estimate the sojourn time, we have made an approximation by supposing that the corresponding distribution follows a negative exponential distribution of parameter γ_{s_k}. Then, the Laplace transform in frequency domain of the sojourn time distribution is

$$e_{s_k} = \frac{\gamma_{s_k}}{\gamma_{s_k} + s} \tag{5.2}$$

Once obtained A, we proceed to compute the vector $\vec{A}^r_{s_k,s_f}$ representing the delay distribution of all connected paths of length r, $r = \{1, 2, 3, ...\}$ (path within the Markov chain) , from state s_k to the final state s_f which is computed as follows:

$$\vec{A}^r_{s_k,s_f} = A \cdot \vec{A}^{r-1}_{s_k,s_f} \tag{5.3}$$

where $\vec{A}^1_{s_k,s_f}$ is the vector containing the delay distribution in one-hop (r = 1) from state s_k to the final one. This vector is a non null vector since there is always one or more states directly connected to the final state. Being s_i the initial state representing the packet arrival event (initial state previously defined), when we look at $A^r_{s_i,s_f}$ we find the delay distribution in r hops from s_i to s_f. Then, the one hop delay distribution in the frequency domain can be computed as follows:

$$D_{f-dom}(s) = \sum_{r=1} A^r_{s_i,s_f} \tag{5.4}$$

The first order derivative of 5.4 evaluated in $s = 0$ will give us the average delay from s_i to s_f. The one hop delay distribution in time domain can be computed by means of the Inverse Laplace Transform (ILP)

$$D_{t-dom}(t) = ILP(D_{f-dom}(s)) \tag{5.5}$$

[1] Delay from the moment the system enters into a particular state until the moment the system changes to another one.

In order to compute the e2e delay from a given node n_{src} to the destination n_{dst} (sink) we need to find the one hop delay distribution in frecuency domain for each intermediate node n_k along the paths from n_{src} to n_{dst}. In case the path is a simple path (Figure 5.3), the e2e delay distribution in frequency domain can be computed as the product of the individual one hop delay for each intermediate node n_k in the path from n_{src} to n_{dst}

$$D_{e2e(f-dom)}(s) = \prod_{i=src}^{y} D_{f-dom}^{(n_i)}(s) \tag{5.6}$$

where n_y is the last hop node before reaching n_{dst} and $D_{f-dom}^{(n_k)}(s)$ the one hop delay in frequency domain for node n_k. Once again, the derivative of the equation 5.6 and its evaluation at $s = 0$ will give us the average end to end delay from source to destination. In case the path is a multi path (Figure 5.3) we can follow the idea from [He et al., 2010b] for the parallel case (equation 2) computing the e2e delay distribution as follows:

$$D_{e2e(f-dom)}(s) = \prod_{i=src}^{j} D_{f-dom}^{(n_i)}(s) \cdot \left(\sum_{i=k}^{s} p_{j,i} \cdot D_{f-dom}^{(n_i)}(s) \right) \cdot \prod_{i=t}^{y} D_{f-dom}^{(n_i)}(s) \tag{5.7}$$

where p_{ki} is the probability for a packet arriving to n_k to be forwarded to node n_i. Finally, the whole e2e delay distribution in time domain can be computed as follows:

$$D_{e2e(t-dom)}(t) = ILP(D_{e2e(f-dom)}(s)) \tag{5.8}$$

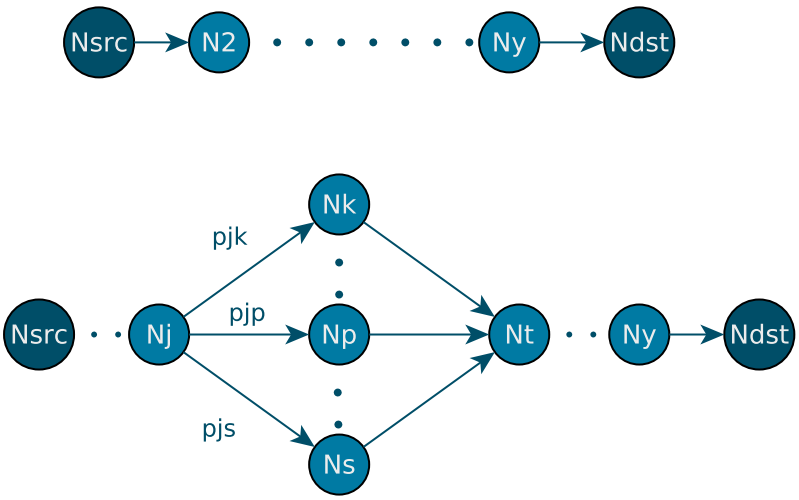


Figure 5.3: Serial (in tandem) and parallel structures.

This mathematical approach is the core of our methodology for estimating the end to end delay in a multi-tranmission scenario. We make use of this in next chapters to find a mathematical expression of the end to end delay distribution.
In next sections we present three contributions where we show how our methodology presented in 5.2 is accurate for modelling the behavior of the wireless sensor network.

5.3 IEEE 802.15.4 MAC Protocol Use Case

In this section, we show how the approach presented in 5.2 can be applied for a well-known MAC protocol. Probability density function for the end to end delay is found for a simple multi-hop transmission scenario.

5.3.1 States Identification

As we said before, this step is one of the most important steps of our approach. The IEEE 802.15.4 MAC protocol has been extensively studied in the literature and a complete specification can be found in [802, 2007]. Hence, the states and transitions can be easily identified from this specification. Figure 5.4 shows the main flow diagram of the standard protocol. For further explanation concerning each variable in the algorithm, please refers to chapter 2, section 2.4. By

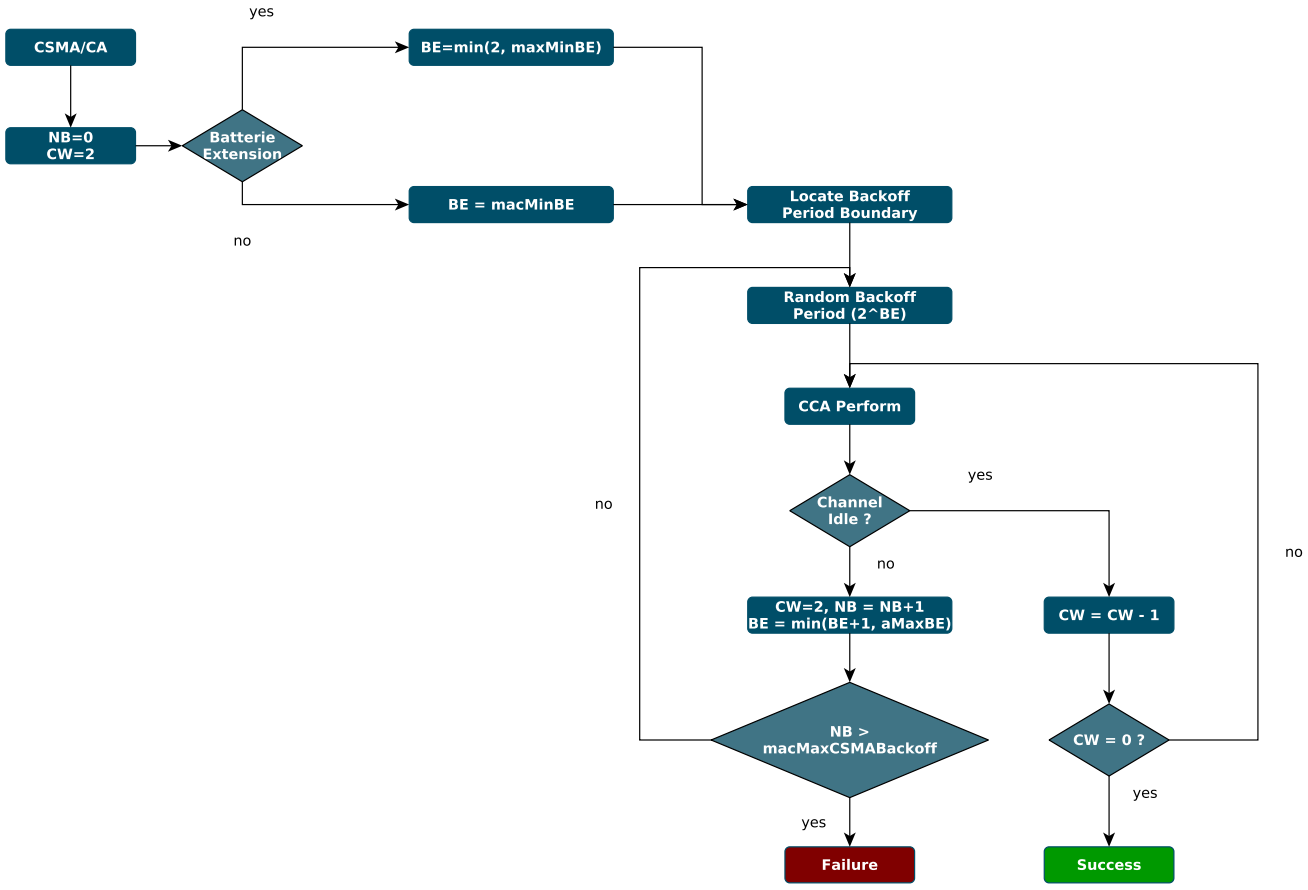


Figure 5.4: IEEE 802.15.4 MAC protocol (slotted version): Flow Diagram.

means of the protocol's specification we were able to identify the following set of states
$\mathbb{S}$ ={ENQUEUING, START_BACKOFF, CCA_1, CCA_2, COLLISION, CHANNEL_BUSY, ACK_RECEIVED}
where

- ENQUEUING: represents the moment a packet arrives to the MAC buffer.

- START_BACKOFF: random backoff while attempting to access the channel.
- CCA_1 and CCA_2: first and second channel assessment.
- COLLISION: when a collision was detected.
- CHANNEL_BUSY: when one of the CCA finds the channel busy.
- ACK_RECEIVED: represents the moment when a packet was succesfully received in the receiver side.

5.3.2 Experiments & Results

Scenario configuration

In order to carry out the experimentation, we have set a testbed with TelosB motes, TinyOS as the underlying operating system and the TKN154 IEEE 802.15.4 MAC implementation. We consider a tree topology as shown in Figure 5.5 where devices (Dx) periodically send unicast packets to a specific router node (Rx). Once receiving a packet, the router forwards it to the coordinator (C). We set three scenarios by varying the Poisson arrival rate to each device: 1, 5 and 10 packets per second. The packet size is set to 32 bytes (21 bytes of payload + 11 bytes of header and trailer) and the buffer length on each node was set to 4 packets. The MAC protocol parameters are the ones by default with a specific duty cycle of 50% ($BO = 6$ and $SO = 5$). The transmission power of each node is 0dBm and the distance between device and router, as well as the one between router and coordinator was set to 1 meter. We have obtained the protocol traces by executing the experiments during five minutes.

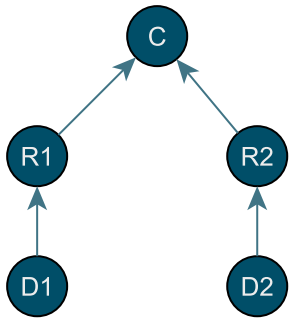


Figure 5.5: TelosB scenario.

Resulting Markov chain

In order to obtain the Markov chain model, we apply the methodology described in the previous section to the set of nodes of the network in the network. Since we are interested in measuring the end to end delay from a given device to the coordinator, we focus our attention on a particular branch of the topology, for instance, the D1-R1-C branch. Then, we proceed to obtain both device and router logs and by means of the procedure described before, we get the corresponding Markov chain model for each node in the path. This Markov chain is defined as follows: let $s(t)$, $b(t)$ and $r(t)$ be the stochastic processes representing the current state, backoff stage and the state of the retransmission counter at time t experienced by a node to transmit a packet,respectively.

Then, $(s(t), b(t), r(t))$ is a three-dimensional Markov chain modelling the MAC protocol behavior. Figure 5.6 presents the Markov chain result for the device D1. A similar result with some differences in states and transition probabilities was obtained for the router R1 node. Two subscripts were added to each state in order to represent the state of retransmissions and the backoff stage. Then, each state in the Markov chain has the S_k_l format where S $\in \mathbb{S}$, k represents the current number of retransmissions and l the number of times the channel was found busy (backoff stage). For instance, being in state $CCA2\_0\_0$, if a collision occurs then we increment the k variable and the Markov chain moves to the $START\ BACKOFF\_1\_0$ to retry the channel assessment. For states PACKET ARRIVAL AND ENQUEUING, states, we omit the subscripts since for these cases, k and l are equal to zero. Figure 5.7 shows the probability

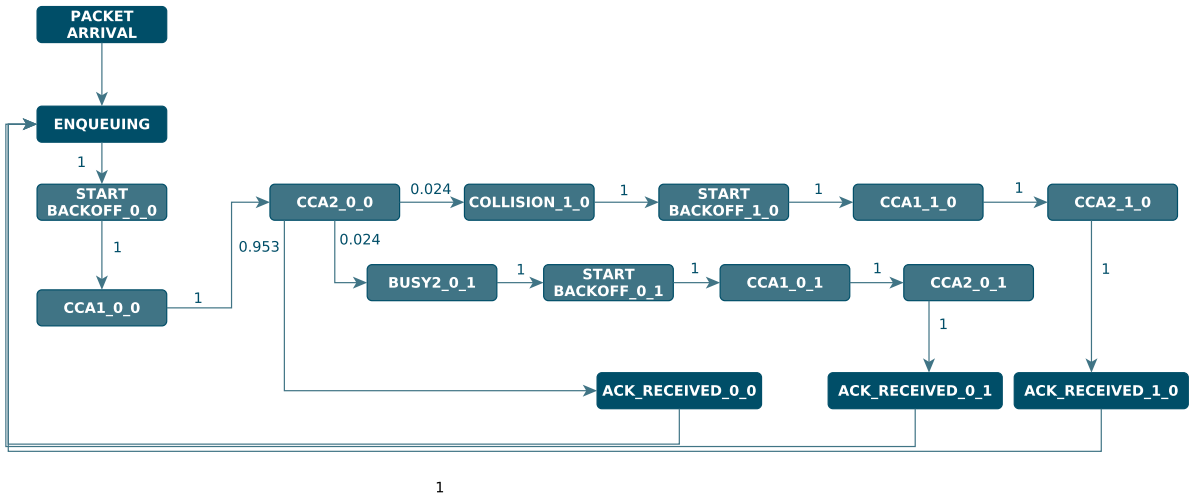


Figure 5.6: Obtained Markov chain of a single device.

density function of the end to end delay from D1 to C computed by the computation module for each scenario. Table 5.1 shows the comparision in terms of the average delay between the measured end to end delay and the one computed by means of our computation module.

λ	Empirical Av. Delay (sec)			Computed Av. Delay (sec)		
	Device	Router	e2e	Device	Router	e2e
1 p/sec	0.1577	0.4481	0.605	0.1578	0.4483	0.606
5 p/sec	0.22	0.498	0.718	0.22	0.498	0.719
10 p/sec	0.345	0.484	0.83	0.345	0.484	0.83

Table 5.1: Empirical and computed e2e average delay.

λ	% Packet dropped	
	Device	Router
1 p/sec	0	0
5 p/sec	12	0
10 p/sec	50	0

Table 5.2: Buffer drop rate.

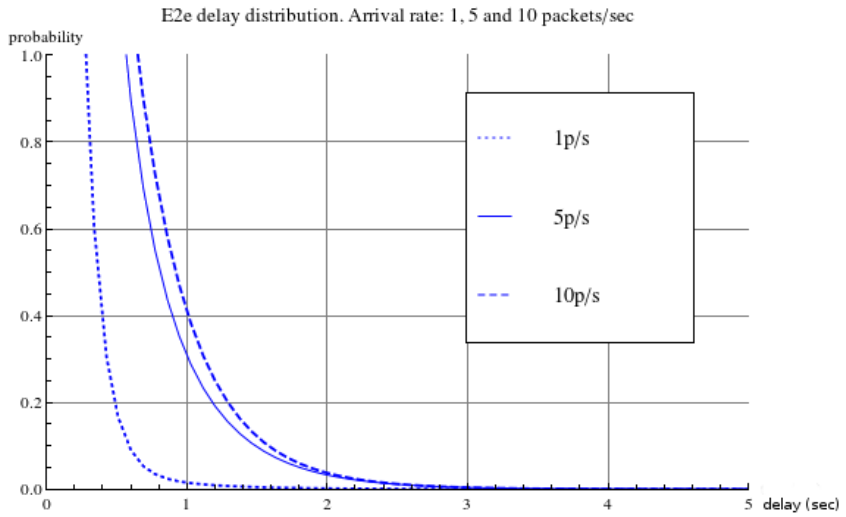


Figure 5.7: Probability distribution function of e2e delay for packet arrival rate $\lambda = 1, 5, 10$ p/s.

From Figure 5.7 we can see that, as the arrival rate increases, the e2e delay also increases. This is due to the fact that, for low traffic scenarios ($\lambda = 1$ p/s) the buffer is almost empty all the time and then queuing delay is minimum. We can see from Table 5.2 that, for the low traffic scenario none of the generated packets were dropped due to buffer overflow for both device and router. On the other hand, as the traffic rate increases, the number of packets dropped during the execution for both $\lambda = 5$ p/s and $\lambda = 10$ p/s also increases. Therefore, the queueing delay will also increases and thus the whole e2e delay.

Discussions

In this section, we have obtained a Markov chain model from the traces of the IEEE 802.15.4 MAC protocol. We have shown that with our approach it is possible to find a realistic model which takes into account many aspects not covered by existing theoretical models. We applied our approach for a tree topology and we have estimated the e2e delay in two hops from the empirical Markov chain and our end to end delay computation module based on the framework presented in [He et al., 2010b]. From results,we can see that our approach is suitable for estimating the e2e delay in a multi-hop transmission scenario, a limitation of most of the existing theoretical models. It is necessary to mention here the limitations of our approach. One of the most important points in the methodology is the one related to the states identification. Therefore, it is necessary to deeply analyse the protocol specification in order to identify all the existing states and transitions and to avoid missing information. This is not a trivial task and means that we should have a comprehensive knowledge of the protocol behavior. Another limitation of our approach, and contrarily to other theoretical models, is the fact that the obtained Markov chain models depend strictly on the input parameters, specially on the packet arrival rate. By changing the input arrival rate we will obtain another Markov chain model where, for instance, transition probabilities will differ from the ones found for some other arrival rate configuration. Further in this thesis, we propose an approach based on non-linear regression to by pass this problem and find a general Markov chain in terms of the packet arrival rate λ, where transitions are not probabilities but functions of λ.

5.4 ContikiMAC protocol Use Case

In this section, we extend previous work by considering another well-known MAC protocol. The idea is to show the suitability of this approach for modelling the behavior of a wireless sensor network independently of the underlying MAC protocol. We present then the approach applied to the ContikiMAC protocol and we present also the results in terms of the probability density function of the end to end delay in a simple multi-hop transmission scenario.

5.4.1 States Identification

In order to indentify the states of the protocol we took a look at the ContikiMAC specification as we have done for the IEEE 802.15.4. In this way, we were able to generate the corresponding flow diagram ilustrating the main states and transition within the protocol. Flow diagram is shown in Figure 5.8. This flow diagram shows both sides of the system, at the top we can see the state diagram from the receiver side while the transmitter side is shown at the bottom of it. The specification, together with the flow diagram, allowed us to identify the relevant states of the protocol such as channel assessment, random backoff, acknowledgement received, sending packet, sending burst, etc. Details on ContikiMAC operation can be found in chapter 2, section 2.5.2.

5.4.2 Results & Discussions

Scenario Configuration

In order to carry out the experimentation we have set a testbed running TelosB motes and ContikiOS version 2.6. We consider an in-tandem topology as seen in Figure 5.9. We have chosen a small example in order to ilustrate our methodology. However, it can be easily extended to more complex scenarios. The distance between device-router and router-coordinator is set to one meter. Both device and router were configured with a buffer length of four packets. Packets are generated at the device as unicast packets and are sent to the router which will then forward them to the coordinator. We set three scenarios by varying the Poisson arrival rate to the device: 1, 2 and 10 packets per second. The packet size is set to 77 bytes (60 bytes of payload + 17 bytes of header). Both router and coordinator send the corresponding acknowledgement once receiving a packet from device and router respectively.

Resulting Markov chain

From subsection 5.4.1, a set of states $\mathbb{S}$ were identified. Looking at the transmitter side of Figure 5.8, the set $\in \mathbb{S}$ is determined by the following states: {*PACKET ARRIVAL, ENQUEUING, CCA1, CCA2, CCA3, CCA4, CCA5, CCA6, SLEEP1, SLEEP2, SLEEP3, SLEEP4, SLEEP5, SLEEP6, CHANNEL_BUSY, SENDING, ACK_RECEIVED, COLLISION, PHASE LOCK, SLEEP BURST, SENDING BURST*}.
Let $s(t)$, $b(t)$ and $r(t)$ be the stochastic processes representing the current state, backoff stage and the state of the retransmission counter at time t experienced by a node to transmit a packet,respectively. Then, $(s(t), b(t), c(t))$ is a three-dimensional Markov chain modelling the behavior of the ContikiMAC protocol. Each state in the Markov chain has the S_k_l format where $S \in \mathbb{S}$, k represents the current number of retransmissions and l the number of times the channel was found busy (backoff stage). Figure 5.10 shows the resulting Markov chain for the device for the scenario $\lambda = 1$. Subscripts for states *PACKET ARRIVAL, ENQUEUING,*

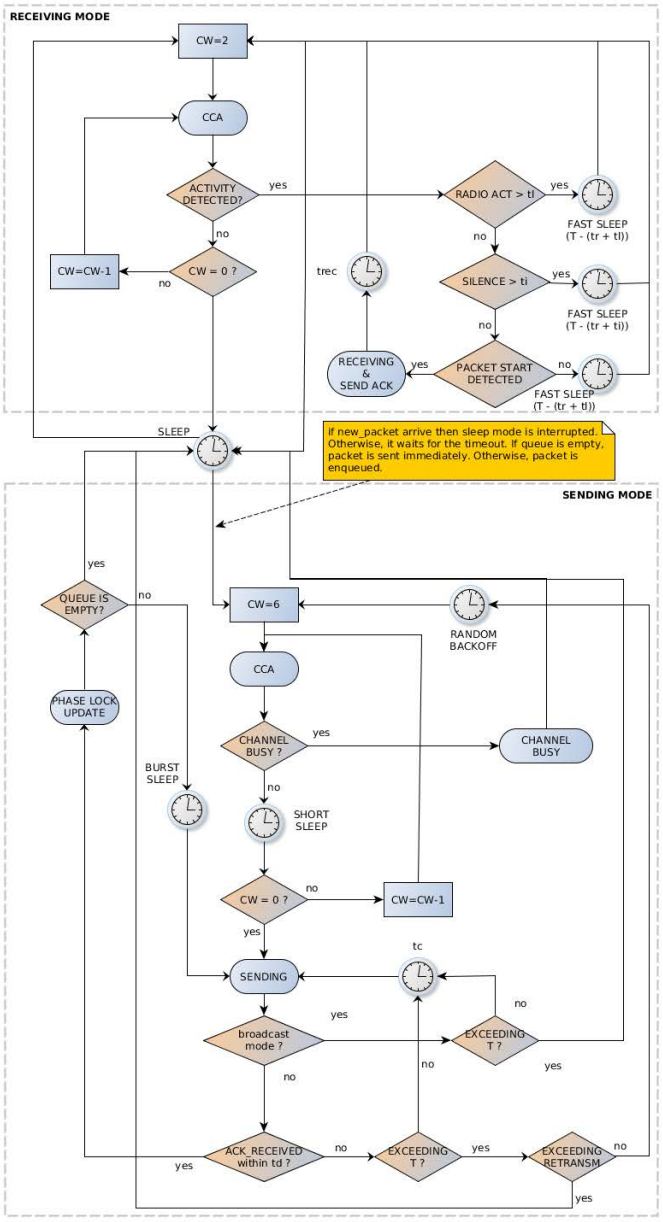


Figure 5.8: ContikiMAC flow diagram.

Figure 5.9: Scenario setup using TelosB nodes and ContikiOS, version 2.6.

SLEEP BURST, *SENDING BURST*, were omitted since, for these cases, k and l are equal to zero. We also omitted the subscripts for the *ACK RECEIVED* state in order to simplify the figure. Here, the square blocks represent the sequence CCA1→SLEEP→CCA2→SLEEP→ $\cdots$ →CCA6→SLEEP→SENDING.

A dotted line between two states S_1 and S_n in a block means that all transitions between intermediate states are equals to 1. Otherwise, if some of the transitions are lower than 1 (meaning that there is a transition to the CHANNEL BUSY state), then we show the state together with the transition probabilities to the next states. Table 5.3 shows the comparision between the

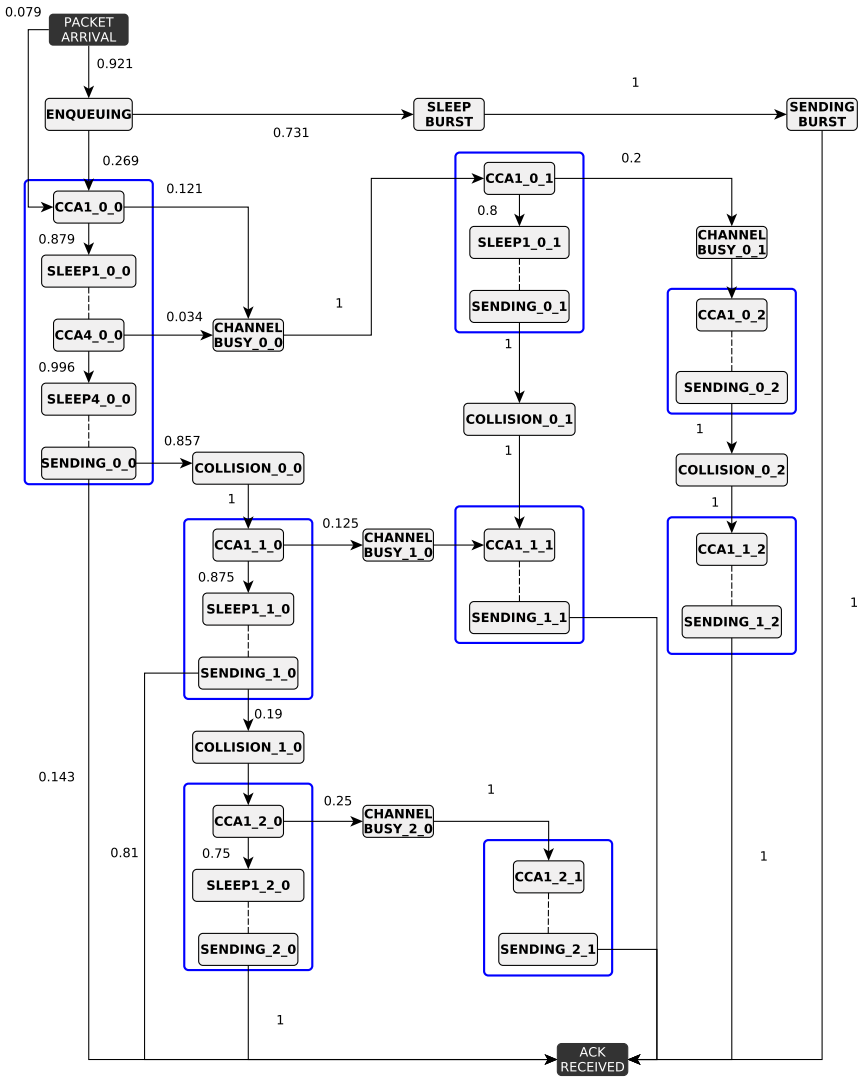


Figure 5.10: Obtained Markov chain for a single device.

measured e2e delay and the one computed by the end to end computation module for the three defined scenarios. Results show that for the three scenarios the computed e2e delay is almost the same as the one measured from the traces of the protocol showing the suitability of the approach for estimating the e2e delay. Figure 5.11 shows the probability density function for the three scenarios. As we can see, as the arrival rate increases the e2e delay also increases. This is due to the fact that for low traffic rate ($\lambda = 1$ packets per second), the queuing delay is almost nonexistent. On the other hand, when considering an arrival rate of $\lambda = 10$ packets per second, the e2e delay increases due to the queuing delay since packets arrive at a high rate and must be stored in the buffer until the packet currently in process is sent to the next hop. To ilustrate this we can take a look at Table 5.4 where we show the percentage of packets dropped due to

λ	Empirical Av. Delay (sec)			Computed Av. Delay (sec)		
	Device	Router	e2e	Device	Router	e2e
1 p/sec	0.11	0.125	0.236	0.12	0.13	0.25
2 p/sec	0.166	0.266	0.432	0.181	0.264	0.44
10 p/sec	0.235	0.38	0.615	0.241	0.383	0.62

Table 5.3: Empirical and computed e2e average delay.

λ	% Packet dropped	
	Device	Router
1 p/sec	0	0
2 p/sec	5	13.7
10 p/sec	10.5	52.7

Table 5.4: Buffer drop rate.

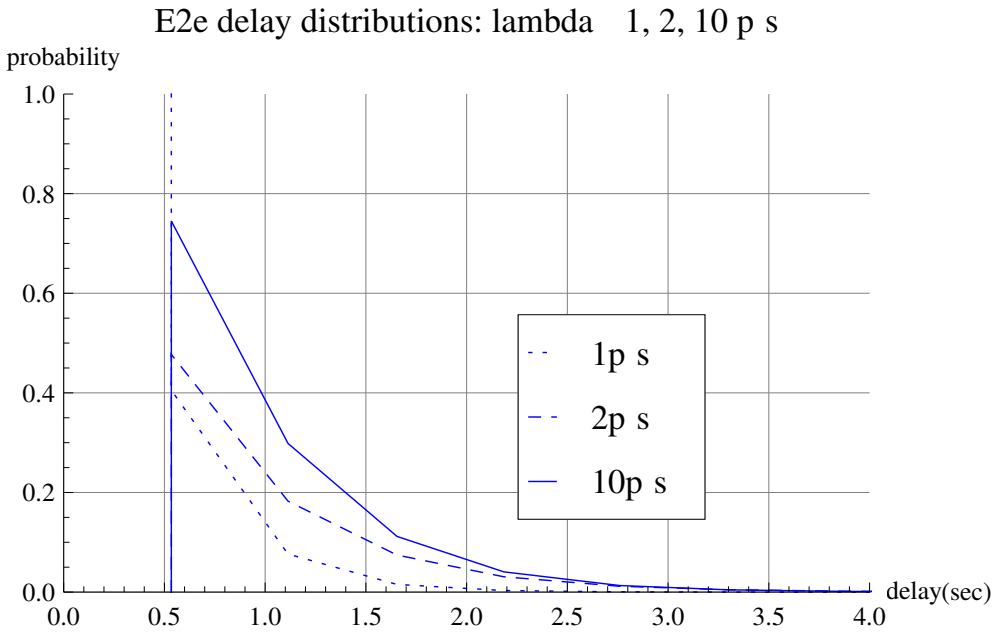


Figure 5.11: Probability density function of e2e delay for packet arrival rate $\lambda = 1, 2, 10$ p/s.

the fact that the buffer was full. As we can see, for $\lambda = 1$ there are no dropped packets meaning that each time a packet arrived to the buffer, either it was empty or with enough capacity to store it. Then, the queuing delay in this scenario has a low impact in the whole e2e delay. On the other hand, when considering $\lambda = 10$ packets per second, we can see that 10% and 53% of packets were dropped at the device and router respectively. That means that, at some moments during the execution of the protocol, both buffers were full and thus the device queueing delay in this case is not negligible and will impact in the whole e2e delay, as seen in Figure 5.11.

5.5 X-MAC & RPL protocols Use Case

Previous work attempted to show the suitability of our approach for modelling a wireless sensor network. However, both contributions consider only static routing where topology is defined at the begining of the experimentation and does not change during it. In this section we extend our analysis by considering also a dynamic routing protocol (RPL) where nodes choose the best path dynamically during the execution of the experimentation. The idea then is to show how our approach can also be considered in a more realistic scenario by considering both the RPL and X-MAC protocols. Besides, we present a comparision in terms of end to end delays for two different routing strategies proposed by the RPL protocol.

5.5.1 States Identification

In this work we have made use of the Contiki implementation of X-MAC protocol (CX-MAC). In order to determine the states of the protocol, we took a look at both the specification and implementation in Contiki of the X-MAC protocol. In this way, we were able to develop the corresponding flow diagram showing each of the states and transitions of the protocol. Figure 5.12 shows the corresponding flow diagram. When a packet arrives (PACKET ARRIVAL) to

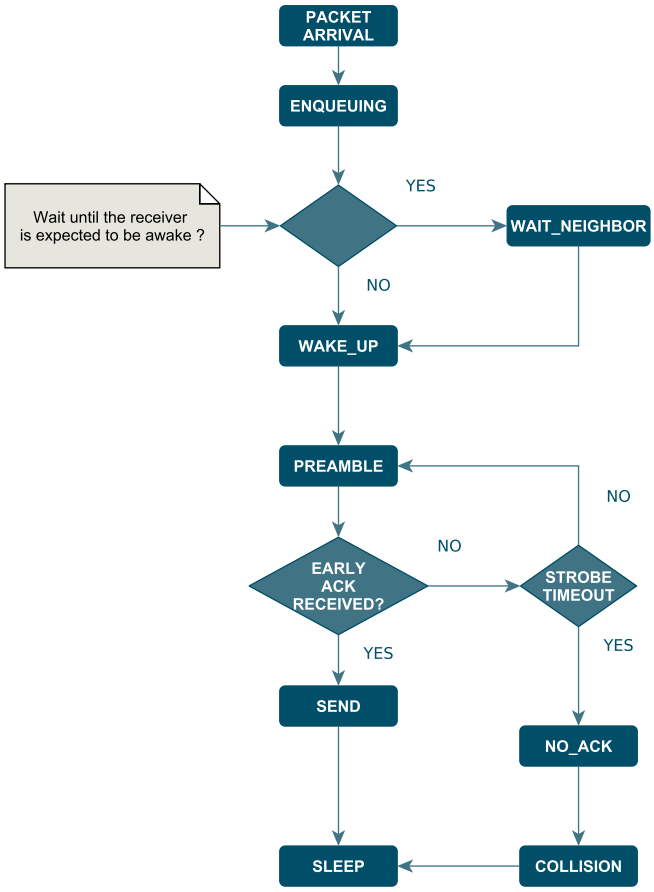


Figure 5.12: State's diagram of X-MAC protocol.

a sender node it would be enqueued (ENQUEUING) in the node buffer and will wait in the

buffer until previous existing packets in the buffer are processed. Before waking up, the node goes through the list of encounters to find if it has recorded an encounter with the receiver neighbor. If so, the node would wait until the receiver is expected to be awake (switching to WAIT_NEIGHBOR) and then it wakes up. Otherwise, the sender will switch immediately to WAKE_UP state. Once the sender has waked up it will starts to send preambles (PREAMBLE). If between two preambles the sender receives an early ack it will send the packet (SEND) and then goes to sleep (SLEEP). Otherwise, it will continue sending preambles until it receives an early ack or until timeout for sending preambles is reached (NO_ACK) switching then to COLLISION state, which basically throws a notification to the upper layer and then go to sleep mode (SLEEP).

5.5.2 Scenario Configuration

We have ran our experimentation in a large-scale infrastructure suitable for testing wireless sensor devices and heterogeneous communicating objects provided by IoT-LAB [Fambon et al., 2014]. Figure 5.13 shows the configuration of the testbed located at INRIA Rennes with 256 WSN430 open nodes. Different from the work done in [Despaux et al., 2014b], [Despaux et al., 2014a] where we have used static routing scenario with TelosB nodes, we have extended our experimentation to consider a dynamic routing provided by the RPL protocol and relies on the implementations of the protocols CX-MAC and RPL available in the Contiki operating system. A total of seven nodes from the testbed were selected to carry out the experiments. Distances between them are shown in Table 5.5. Here, node 145 was configured as the sink node while the others generate packets with a given inter-arrival rate and send them to the sink. The scenarios were

	145	205	236	166	45	14	54
145	*	4.16	9.97	10.03	12.53	9.16	4.5
205	-	*	7.89	9.39	13.81	11.47	7.91
236	-	-	*	3.51	9.82	10.24	10.46
166	-	-	-	*	6.32	7.29	8.93
45	-	-	-	-	*	4.30	9.09
14	-	-	-	-	-	*	5.1
54	-	-	-	-	-	-	*

Table 5.5: Distances (meters) between selected nodes.

divided in two groups, the first group uses the RPL objective function 0 (OF0) as a routing metric and the second one uses the ETX objective function. For each of these groups, a set of four scenarios with different arrival rates $\lambda = 0.5p/s$, $\lambda = 1p/s$, $\lambda = 2p/s$, $\lambda = 4p/s$ were defined. The packet size was defined as 42 bytes (25 bytes of payload + 17 bytes of header) and the buffer length on each node was set to 8 packets. Power transmission on each node was set to 0dBm. The simulation time for $\lambda = 0.5$ and $\lambda = 1$ was set to 16 minutes while for $\lambda = 2$ and $\lambda = 4$ was 10 minutes. It is important to note that we have specifically selected a high traffic load scenario ($\lambda = 4$) in order to test the performance of our approach under a saturated scenario. For the sake of simplicity and in order to avoid having a huge Markov model, we do not consider retransmissions in our experimentations. However, the approach is still valid if we consider also retransmissions.

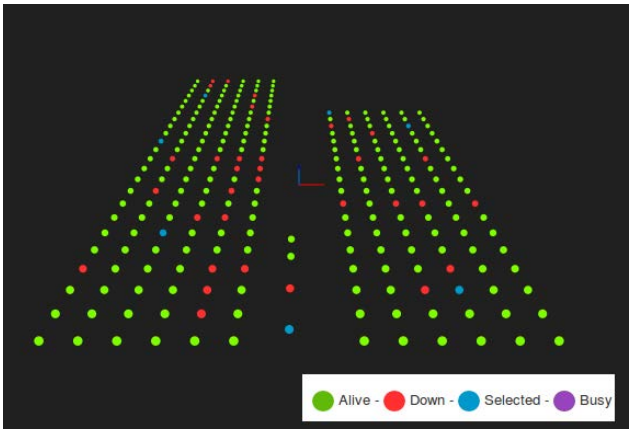


Figure 5.13: IOT-LAB Testbed at Rennes/France.

5.5.3 Results and Discussion

Resulting Markov Chain

Previously in section 5.5.1, a set $\mathbb{S}$ of states has been identified. This set is determined by the states $\mathbb{S}$={*PACKET ARRIVAL ENQUEUED, WAIT NEIGHBOR, WAKE UP, PREAMBLE, SLEEP, ACK RECEIVED, SEND*}. As we have seen in chapter 2 section 2.5.2, the protocol X-MAC use short strobed preambles which embeds information regarding the destination address. Differently from other protocols, X-MAC insert small pauses between short preambles during the time within which the sender listens to the medium. During this time, receiver can send an early acknowledgement packet back to the sender. Once the sender receives an acknowledgement from the receiver it stops sending preambles and sends the data packet. Let $s(t)$ and $p(t)$ be the stochastic processes representing the current state and the preamble stage (the number of preambles sent without receiving an early acknowledgement). Different from previous contributions, we do not consider retransmissions for this case. In this way, the $(s(t), p(t))$ is a bi-dimensional Markov chain representing the behavior of the CX-MAC protocol. Figure 5.14 shows the Markov chain model obtained for node 166. Each state in the Markov chain has the S_k format where $S \in \mathbb{S}$ and k represents the number of preambles sent without receiving an early acknowledgement (preamble stage). Since the subscript k only affects the state *PREAMBLE*, we omit it in the rest of the states. It is important to note that we have obtained a Markov chain for each node in the experimentation. In Figure 5.14 we can see basically the states and transitions that we have identified previously in section 5.5.1 together with the probability of transitions from each state to the other ones. Two other states were added *(BUFFER FULL and PACKET DROPPED)* in order to represent the packets dropped due to buffer full and the fact of reaching the maximum number of preambles sent. An important thing to note is that this Markov chain represents the behavior in one node and would be helpful for estimating the delay in one hop for such node. Hence, this Markov chain represents the behavior of the CX_MAX protocol running locally in the node. On the other hand, RPL protocol works over the MAC layer and thus its scope covers the totality of the network (by determining the best path from one node to the sink, assigning node parents, etc). Hence, and in order to estimate the one hop delay in one node, it is enough to have a Markov chain modelling the behavior of the underlying MAC protocol.

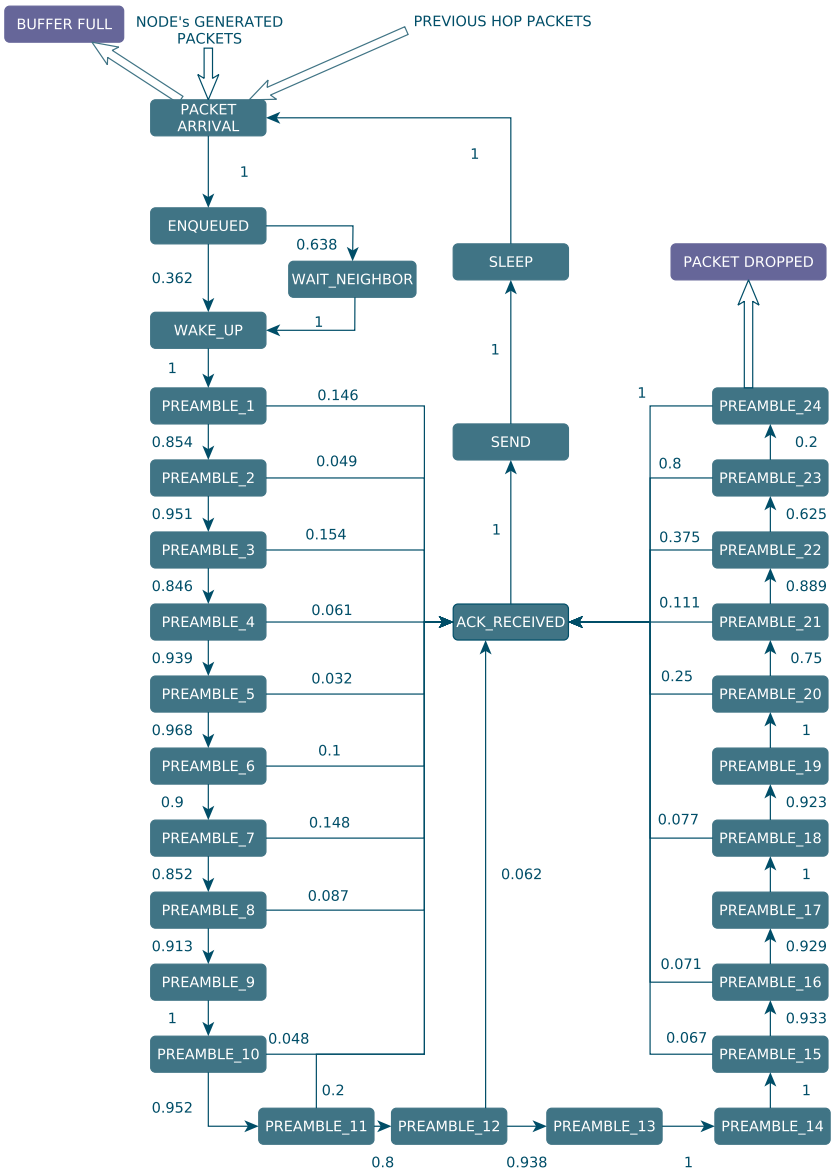
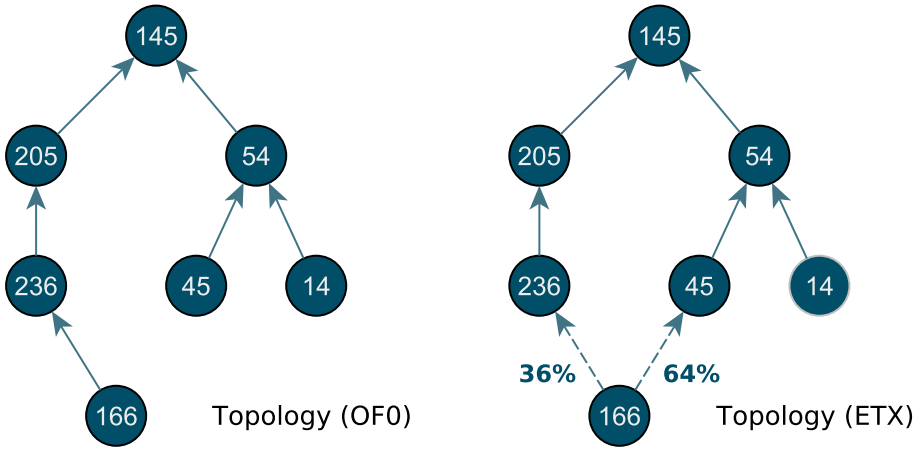


Figure 5.14: Markov chain model for node 166 ($\lambda = 4$).

One-hop and End to End delay

As we mentioned before, experimentations were done by defining two sets of scenarios, the first considering the OF0 RPL objective function and the second one considering the ETX metric. Then, for each subset we executed the experimentation considering four different arrival rates values. Figures (5.16a) and (5.16b) show the routing graphs (DODAGs) generated by the RPL protocol for both objective functions during the execution of the experimentation. In the case of the ETX objective function scenario, for high traffic load ($\lambda = 4$), the parent selection mechanism of node 166 fluctuates between nodes 45 and 236 in such a way that, at the end of the execution, 64% of the generated packets were sent to the sink through node 45 while the rest 36% of the packets were sent through node 236. Tables 5.6 and 5.7 present, for each node, the average one-

hop and e2e delay empirically measured together with the theoretical average delay estimation for both one-hop and e2e obtained by means of the methodology described in the previous section. The empirical average delays were obtained by measuring the delay for each generated packet. In order to be able to make a comparison in terms of delay between the two objective functions, we have computed the global average delays. For each scenario (objective function/λ) we compute the average one-hop delay among all nodes in the network. The same method is applied to the e2e average delay for each scenario. A summary of these results is shown in Tables 5.8 and 5.9. Then, we were able to compare both metrics by looking at the global one-hop and e2e delay results. Figure 5.17 shows the probability density function of the e2e delay from node 166 to the sink (145) for both RPL objective functions. The e2e delay distribution was obtained by means of the analysis done in section 5.2.3. We only consider the high traffic load scenario ($\lambda = 4$) to plot since it is the scenario where results for both objective functions are more significantly different. Finally, Table 5.10 shows the percentage of the Packet Reception Rate (PRR) for each objective function.



(a) OF0 topology ($\lambda = 0.5, 1, 2, 4$). ETX topology ($\lambda = 0.5, 1, 2$)

(b) ETX topology. ($\lambda = 4$).

Figure 5.15: Routing graphs for ETX and OF0 objective functions.

ETX	Emp. Global Av. Delay		Theo. Global Av. Delay	
λ	1-hop	e2e	1-hop	e2e
0.5	0,1503	0,2497	0,1482	0,2516
1	0,1467	0,2427	0,1472	0,2530
2	0,1613	0,2605	0,1569	0,2717
4	0,1685	0,2981	0,1627	0,2870

Table 5.9: Empirical and theoretical global average delay for each λ (ETX).

OF0	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 0.5$	205	0,1282	0,1282	205	0,1293	0,1293
	14	0,1417	0,2178	14	0,1430	0,2626
	54	0,1192	0,1192	54	0,1197	0,1197
	45	0,1561	0,2768	45	0,1550	0,2747
	166	0,1384	0,4272	166	0,1371	0,4275
	236	0,1618	0,2747	236	0,1611	0,2904
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 1$	205	0,1193	0,1193	205	0,1208	0,1208
	14	0,1634	0,2625	14	0,1597	0,2851
	54	0,1251	0,1251	54	0,1254	0,1254
	45	0,1394	0,2396	45	0,1418	0,2672
	166	0,1920	0,4505	166	0,1822	0,4657
	236	0,1651	0,2663	236	0,1627	0,2835
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 2$	205	0,1278	0,1278	205	0,1278	0,1278
	14	0,1851	0,2949	14	0,1806	0,3066
	54	0,1277	0,1277	54	0,1260	0,1260
	45	0,1706	0,2869	45	0,1677	0,2937
	166	0,1503	0,3917	166	0,1406	0,4366
	236	0,1720	0,2898	236	0,1681	0,2959
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 4$	205	0,1428	0,1428	205	0,1432	0,1432
	14	0,2127	0,3948	14	0,1970	0,3668
	54	0,1635	0,1635	54	0,1697	0,1697
	45	0,2027	0,3737	45	0,2052	0,3749
	166	0,2055	0,4835	166	0,2055	0,5359
	236	0,1915	0,3553	236	0,1870	0,3303

Table 5.6: Empirical and theoretical e2e average delay (OF0).

	OF0	ETX
λ	PRR (%)	PRR (%)
0.5	97	98
1	96	95
2	33	35
4	2.8	9.9

Table 5.10: Packet reception rate (PRR) for OF0 and ETX objective functions.

Discussions

Figures (5.16a) and (5.16b) show the DODAGs created by the RPL routing protocol during the experiment. For $\lambda = 0.5$, $\lambda = 1$ and $\lambda = 2$, both metrics share the same topology (5.16a). On the other hand, as traffic rate increases ($\lambda = 4$) and due also to the fact of having a buffer size of eights packets on each node, the quality of the path in terms of expected transmission count between node 166 and the initial assigned parent (45) decreases. Hence, RPL parent selection mechanism for node 166 starts to oscillate between nodes 236 and 45 and this phenomenon is

ETX	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 0.5$	205	0,1123	0,1123	205	0,1111	0,1111
	14	0,1559	0,2655	14	0,1469	0,2656
	54	0,1202	0,1202	54	0,1187	0,1187
	45	0,1656	0,2831	45	0,1652	0,2839
	166	0,1877	0,4503	166	0,1865	0,4585
	236	0,1604	0,2670	236	0,1609	0,2720
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 1$	205	0,1173	0,1173	205	0,1157	0,1157
	14	0,1580	0,2590	14	0,1592	0,2837
	54	0,1269	0,1269	54	0,1245	0,1245
	45	0,1536	0,2798	45	0,1549	0,2793
	166	0,1690	0,4158	166	0,1751	0,4449
	236	0,1552	0,2571	236	0,1542	0,2698
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 2$	205	0,1283	0,1283	205	0,1252	0,1252
	14	0,1886	0,3034	14	0,1821	0,3120
	54	0,1327	0,1327	54	0,1299	0,1299
	45	0,1754	0,2926	45	0,1744	0,3044
	166	0,1639	0,4119	166	0,1518	0,4554
	236	0,1790	0,2939	236	0,1780	0,3033
	Empirical Av. Delay			Theoretical Av. Delay		
	Node	1-hop	e2e	Node	1-hop	e2e
$\lambda = 4$	205	0,1305	0,1305	205	0,1296	0,1296
	14	0,1989	0,3415	14	0,1847	0,3245
	54	0,1423	0,1423	54	0,1398	0,1398
	45	0,2009	0,3367	45	0,1921	0,3320
	166	0,1143	0,4476	166	0,1144	0,4511
	236	0,2240	0,3898	236	0,2155	0,3451

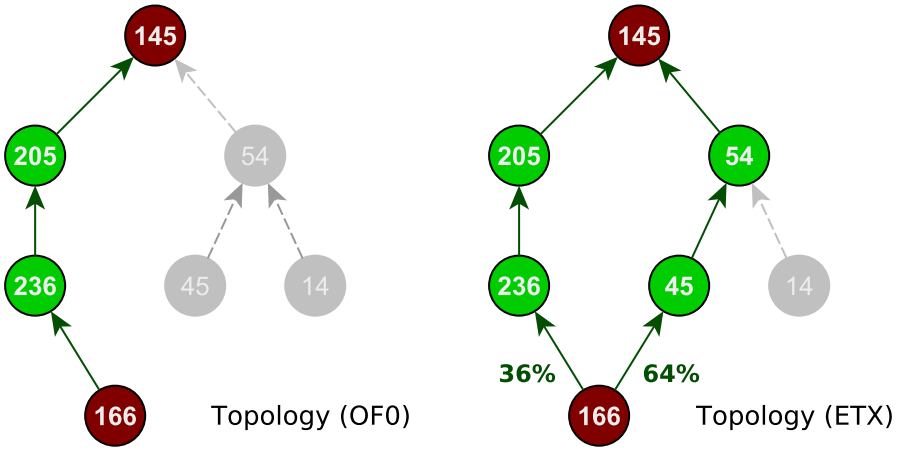
Table 5.7: Empirical and theoretical e2e average delay (ETX).

OF0	Emp. Global Av. Delay		Theo. Global Av. Delay	
λ	1-hop	e2e	1-hop	e2e
0.5	0,1409	0,2406	0,1408	0,2507
1	0,1507	0,2439	0,1488	0,2580
2	0,1556	0,2531	0,1518	0,2644
4	0,1864	0,3189	0,1846	0,3202

Table 5.8: Empirical and theoretical global average delay for each λ (OF0).

repeated time and time again throughout the experiment. This gives an alternative topology (5.16b) for the case of high traffic load. Tables 5.6 and 5.7 shows the empirical and theoretical delay for each node on each scenario. When comparing empirical vs theoretical delays we can see that, for the 1-hop case, empirical and theoretical delays are almost the same with an average difference between them of ≈ 2 milliseconds. For the case of e2e delay estimation, we can see that, for certain nodes, theoretical estimation differs a little bit from empirical measures. However, the average difference is $\approx$ 10 milliseconds and thus, the proposed approach described in the last section remains an acceptable way for estimating the e2e delay in a wireless sensor network.

Tables 5.8 and 5.9 show results in terms of the global average delay for each of the scenarios. The first conclusion we can deduce from this result is that, as traffic load increases, the average delay also increases. This is due to the fact that the network congestion also increases and since the buffer length is limited (8 packets), the time taken for each packet to be dispatched also increases and therefore the e2e delay from the source node to the destination (sink). Concerning the comparison between both metrics, we can see that for low and moderate traffic load ($\lambda = 0.5$, $\lambda = 1$ and $\lambda = 2$), the global average delays are almost the same. On the other hand, when considering a high traffic load ($\lambda = 4$), the global average delay is significantly lower for the ETX case. This is a consequence of the RPL ETX strategy which builds an optimal path in terms of link quality and avoids sending packets over a congested link. The alternative path is supposed to be better in terms of link quality and congestion and thus it should improve network performance in terms of end to end delay. This can be clearly observed from the obtained delays of node 166. We can see from Tables 5.6 and 5.7 that for low and moderate traffic load scenarios, the e2e delay of node 166 does not considerably differ between the two metrics while for the high traffic load scenario, the obtained e2e delay, by means of ETX, is significantly lower than the one obtained by OF0. Figure 5.16 shows the chosen paths for each strategy for the scenario $\lambda = 4$. Based on equations 5.6 and 5.7, we were able to compute the e2e delay distributions from node 166 to the sink (145) for both RPL objective functions. Figure 5.17 show the probability density distribution of the e2e delay. As shown in the plot, the whole e2e delay is significantly lower when considering ETX instead of OF0 objective function. Finally, Table 5.10 shows the percentage of the Packet Reception Rate (PRR) for each objective function. Intuitively, we can expect the PRR for the ETX metric to be greater than the PRR for the OF0 since ETX looks for the best path from source to destination in terms of the expected transmission count. This issue can be confirmed from the results, specially, for the high traffic load scenario where PRR is improved ($\approx 7\%$) when considering the ETX metric.



(a) Chosen path for OF0 strategy ($\lambda = 4$).(b) Chosen path for ETX strategy ($\lambda = 4$).

Figure 5.16: Routing graphs for ETX and OF0 objective functions.

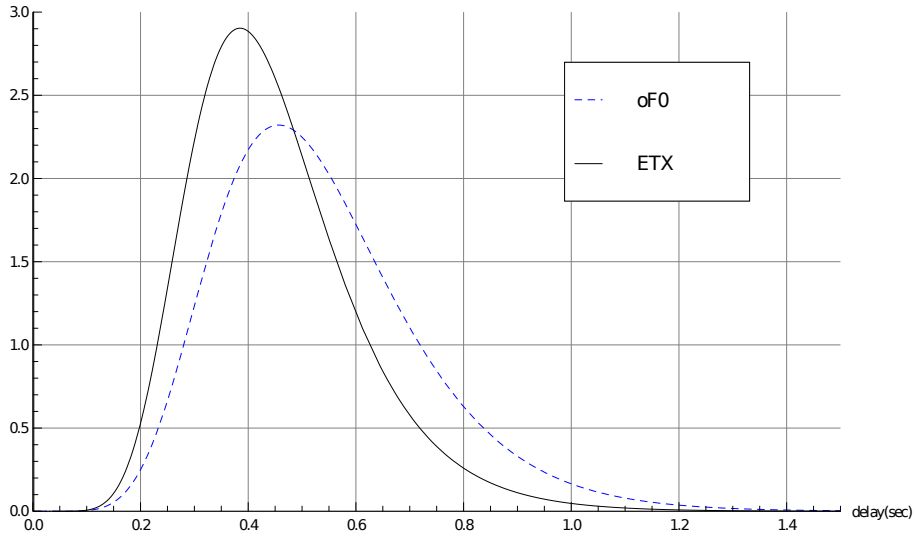


Figure 5.17: Probability density function of the e2e delay from node 166 to the sink (145) for both RPL objective functions (high traffic load scenario).

5.6 Conclusion

In this chapter we have presented an approach to model the behavior of sensor nodes in a WSN using a Markov chain obtained from MAC protocol execution traces. This approach allows us to obtain a Markov chain for each of the nodes presents in the network. We also proposed a mathematical approach for estimating, first, the one hop delay on each node, that is to say, the delay from the moment a packet arrives to the node, until the corresponding acknowledgement is received. The probability density function of the one hop delay is then found and the probability density function of the end to end delay is then computed based on the frequency domain analysis proposed by [He et al., 2010a]. In order to validate the approach we have presented three use cases where we apply our approach for finding the probability distribution function of the end to end delay for three different MAC protocols (IEEE 802.15.4, ContikiMAC, X-MAC) considering both static and dynamic routing (RPL). For the case of dynamic routing, we make use of our approach for comparing two routing strategies of the RPL protocol in terms of the end to end delay. Considering the fact that the end to end delay estimation is computed from individual one-hop delay distributions, we can say that our approach is suitable for estimating the end to end delay independently of the chosen routing protocol. Contrarily to the IEEE 802.15.4 MAC protocol, and in spite of the spread of ContikiMAC, no theoretical model exists in the literature for modelling this protocols. We have also seen that the proposed model for X-MAC [Yang and Heinzelman, 2012] is not enough accurate for modelling its behavior. Hence, we contribute to the literature by presenting a comprehensive Markov model for both protocols. However, these obtained Markov chain models are not general in the sense that they depends on the traffic load. In the next chapter we propose a mathematical approach based on non-linear regression for generalising the inferred Markov chains in terms of the packet arrival rate.

Chapter 6

Methodology Generalisation Using Non-Linear Regression Techniques

Contents

6.1 Introduction

In previous chapter, we have introduced our approach for finding a Markov chain model for modelling the behavior of sensor nodes and, by means of this model, to be able to estimate the end to end delay in a multi-hops scenario. However, the obtained Markov chain model is not general in the sense that it is traffic dependent. That's means that, for differents arrival rates, resulting Markov chains would be different.
In this chapter, we introduce a way for generalizing a Markov chain model in terms of the arrival rate by means of a non-linear regression approach. With this approach, we are able to obtain a generic Markov chain model whose transition's probabilities and state's sojourn time are functions of the packet arrival rate λ. As we said before, there are some other parameters that impacts the Markov chain model such as the number of nodes, the transmission power, etc. However, a similar non-regression approach can be done for these other parameters in order to extend this generalisation. From the obtained model we are able to estimate the end to end delay in a wireless sensor network for any given packet arrival rate λ. In this chapter, and after a

brief background in non-linear regression, we give an introduction to our approach and we show how it can be applied in order to obtain a generic Markov chain model for X-MAC and RPL scenarios.

6.2 Regression Analysis Technique

Regression analysis refers to the statistical inferences for a model

$$y = f(x, \theta) + \epsilon \tag{6.1}$$

where $y \in \mathbb{R}$ is the response variable, $x = (x_1, x_2, \cdots, x_k) \in \mathbb{R}^k$ are explanatory variables and $\theta = (\theta_1, \theta_2, \cdots, \theta_k) \in \mathbb{R}^k$ are parameters. The function f is named *Regression Function* and ϵ is an error term with zero mean and variance σ^2. When the regression function f is linear in the parameters θ, it leads to the very popular and widely used statistical infenrential technique known as linear regression analysis. On the other hand, in the case f is not linear it is necessary to apply a non-linear regression analysis. The objective of non-linear regression analysis is to fit a model to the data minimizing the sum of squared deviations of the dependent variable about the "best fit" curve. Almost any function that can be written in closed form can be incorporated in a non-linear regression model.
Some examples of non-linear models include:

$$f(x, \theta) = \frac{\theta_0 + \theta_1 x}{1 + \theta_2 x} \tag{6.2}$$

$$f(x, \theta) = \theta_1 x^{\theta_2} \tag{6.3}$$

$$f(x, \theta) = \theta_0 + \theta_1 exp(-\theta_2 x) \tag{6.4}$$

$$f(x, \theta) = \theta_1 \sin(\theta_2 + \theta_3 x_1) + \theta_4 \cos(\theta_5 + \theta_6 x_2) \tag{6.5}$$

There are many estimation methods of non-linear regression, one of the most popular is the *least squares* methods that we present in the next subsection.

6.2.1 Least Squares Method

The least squares estimator of θ, denoted by $\hat{\theta}$, is the point in the parameter space such that $f(\hat{\theta})$ is closest to y in the sample space among all feasible $f(\theta)$. In other words, least squares problems minimize the difference between a set of data and a model function f that approximates these data. The least squares estimator is derived from minimization of the residual sum of squares:

$$S(\theta) = \sum_{i=1}^{n} \{y_i - f(x, \theta)\}^2, \quad \theta \in \Omega \subset \mathbb{R}^p. \tag{6.6}$$

By minimizing $S(\theta)$, we can find the parameters that most accurately match the model to the observed data. When f is differentiable with respect to θ, a system of equations needs to be solved for the least square solution θ in the following system of equations:

$$\frac{\partial S(\theta)}{\partial \theta_l}\Big|_{\theta=\hat{\theta}} = 0, \quad l = 1, ..., p. \tag{6.7}$$

The system of equations (called normal equations) are given by

$$\sum_{i=1}^{n} \frac{\partial f(x_i, \theta)}{\partial \theta_l} \{y_i - f(x_i, \theta)\}\Big|_{\theta=\hat{\theta}} = 0 \tag{6.8}$$

Often normal equations do not have an analytic solution for $\hat{\theta}$ and numerical iterative procedures are needed. One of these methods is the well known Gauss-Newton. This method solves the non linear least squares problem based on iterative local linear approximations to the solution locus.

Newton's Method

Newton's method is an algorithm for locating roots that serves as the basis for the Gauss-Newton method derivated from the Taylor series expansion of a function $f(x)$ at a point $x = x_0 + \delta$. Newton's method uses the first-order approximation

$$f(x_0 + \delta) \approx f(x_0) + f'(x_0)\delta \tag{6.9}$$

which is the equation of the tangent line to the curve at an initial guess x_0. The point where this tangent intersects the x-axis will be the next guess x_1 and is given by

$$x_1 = x_0 + \delta_0 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{6.10}$$

Hence, Newton's method is the iterative process

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{6.11}$$

A slightly different version of Newton's method can be used to find the extreme points of a function rather than its roots:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)} \tag{6.12}$$

This formulation serves as the basis of the Gauss-Newton method.

Gauss-Newton Method

The Gauss-Newton method is based on the basic equation from Newton's method except that is uses a search direction vector p_k^{GN} and a step size α_k in the revised equation

$$x_{k+1} = x_k + \alpha_k p_k \tag{6.13}$$

The values that are being altered in this case are the variables of the model function $f(x, y_j)$. Like Newton's method, Gauss-Newton is an iterative process repeating equation (6.13) until the model function fits the data points satisfactorily. For a complete review on this method, see [Deuflhard, 2011].

In the next section, we present the methodology based on non-linear regression and least squares in order to generalize a Markov chain model for estimating the end to end delay independently of the specified packet arrival rates.

6.3 Methodology

In this section we present our methodology for generalizing a Markov chain model, that is to say, to obtain a Markov chain where transition probabilities and state sojourn time are not values but functions of the packet arrival rate. This methodology is based on a non-linear regression technique and the proposed solution is based on the least squares estimation method. As mentioned before, least squares method minimizes the difference between a set of data and a model function f that approximates it. Therefore, our approach consists in generating a set of data in order to be able to estimate this function f.
To generate this data set, and considering the fact that we are focused in finding a Markov chain generalisation in terms of the packet arrival rate λ, the first step is to generate a set of Markov chains for each node for different arrival rates λ. We are going to explain how we proceed for one node keeping in mind that the same approach is done for all nodes belonging to the network. In this way, we define the set of arrival rates $\mathcal{A}_{rate} = \{\lambda_i\}$ where i represents i packets per second. Then, for each λ_i we obtain the corresponding Markov Chain $\mathcal{MC}_{\lambda_i}$ or, in other words, a set of transition's probabilities $T_{\lambda_i} = \{t^{(1)}_{\lambda_i}, t^{(2)}_{\lambda_i}, ..., t^{(K)}_{\lambda_i}\}$ and a set of state's sojourn time[2] $S_{\lambda_i} = \{s^{(1)}_{\lambda_i}, s^{(2)}_{\lambda_i}, ..., s^{(L)}_{\lambda_i}\}$, being $|K|$ and $|L|$ the number of transitions and states present in $\mathcal{MC}_{\lambda_i}$, respectively.
Once generated both T_{λ_i} and S_{λ_i} for each of the λ_i, what we do is to group transitions and states information in order to apply the non-linear regression approach to generalize the Markov chain. This is done as follows: we consider the set $\mathcal{P}_t = \{P^{t^{(1)}}, P^{t^{(2)}}, ..., P^{t^{(K)}}\}$, where $P^{t^{(j)}}$ is the set of transition probabilities for transition $t^{(j)}$ taken from each T_{λ_i}. As an example, given the set $\mathcal{A}_{rate} = \{\lambda_1, \lambda_2, \lambda_3\}$ and the corresponding Markov chains $\mathcal{MC}_{\lambda_1}$, $\mathcal{MC}_{\lambda_2}$, $\mathcal{MC}_{\lambda_3}$, the corresponding $P^{t^{(k)}}$ for transition $t^{(k)}$ is the set of probabilities found in $\mathcal{MC}_{\lambda_1}$, $\mathcal{MC}_{\lambda_2}$, $\mathcal{MC}_{\lambda_3}$ for $t^{(k)}$. In case a particulary transition $t^{(p)}$ previously found in some of the T_{λ_i}, is not present in a particulary set T_{λ_j}, we will consider this probability as zero and we will add this value to the corresponding $P^{t^{(p)}}$. It is worth to note the importance in our approach to identify all the states. Normally, for heavy traffic, we expect to obtain a complete Markov chain in terms of the states and transitions previously identified. In the same way, we consider the set $\mathcal{D}_s = \{D^{s^{(1)}}, D^{s^{(2)}}, ..., D^{s^{(L)}}\}$, $D^{s^{(j)}}$ taken from each S_{λ_i}. In this case, $D^{s^{(j)}}$ does not represent probabilities but the delay spent on state $s^{(j)}$ before transitioning to another one.
Finally, non-linear regression can be applied independently for each element belonging to the set $\mathcal{P}_t$ in such a way that, for each transition $t^{(j)}$, we obtain an approximated function $f_{t^{(j)}}$ that allows us to estimate the transition's probability for an unknown packet arrival rate λ_{uknw}. In the same way, by applying non-linear regression over each element of the set $\mathcal{D}_s$ we obtain a function $f_{s^{(j)}}$ that allows us to estimate the sojourn time on each state.

6.3.1 Applying Non-linear Regression

As said before, $\mathcal{P}_t$ represents a set where each element is also a set. Concretely, each element represents a set of probabilities for a given transition found in the set of T_{λ_i}. The idea then is to apply non-linear regression on each element of the set $\mathcal{P}_t$ in such a way that the output would be a function $f_{t^{(j)}}$ modelling the behavior of transition $t^{(j)}$ $\forall j \in \{1..K\}$. In the same way, a function $f_{s^{(j)}}$ allowing us to estimate the sojourn time on each state can be found by means of this approach.
We start by considering a set of functions useful for estimating both transition's probabilities

[2] The time we spend on one state before transitioning to another one.

and state's sojourn time. This set is not limited and can be extended by adding new functions to it. In our case, we have selected the most common functions detailed below:

- Exponential: $a * \exp(b * x)$
- Linear: $a + b * x$
- Parabolic: $a + b * x + c * x^2$
- Logarithmic: $a + b * \log(x)$
- Sinuosity: $a - (b * \sin(c * x + d) * e * \cos(f * x + g))$

Then, for each set of probabilities $P^{t(j)}$, we apply non-linear regression for each of the functions presented above. In other words, for each of these functions, we estimate the best parameters *(a, b, c, ...)* that best fit the data. In this way, we obtain the best exponential, linear, parabolic, logarithmic and sinuosity functions fitting transition's probabilities. In order to chose the best function f, we analyse the residual sum of squares and we chose the one that minimize

$$\sum_{i=1}^{n} \{y_i - f(x, \theta)\}^2 \tag{6.14}$$

This approach is also done for estimating $f_{s^{(j)}}$ modelling the sojourn time for each of the states. In this way, we obtain a general Markov chain model $\mathcal{GMC}_\lambda$ where both transition's probabilities and state's sojourn time are not values but mathematical functions of λ value as shown in Figure 6.1. By evaluating λ on each function for a given value of λ we obtain the corresponding Markov

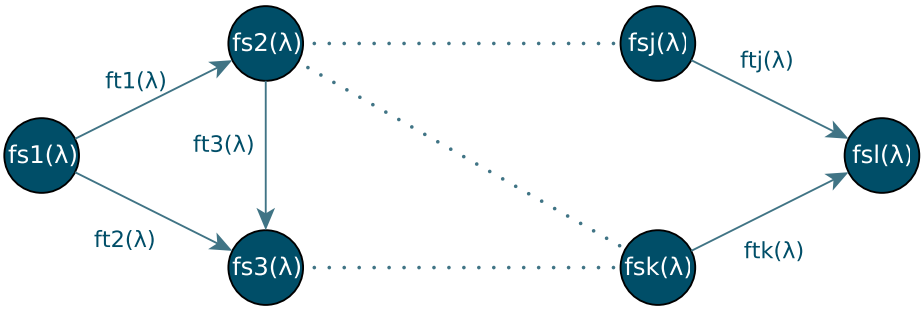
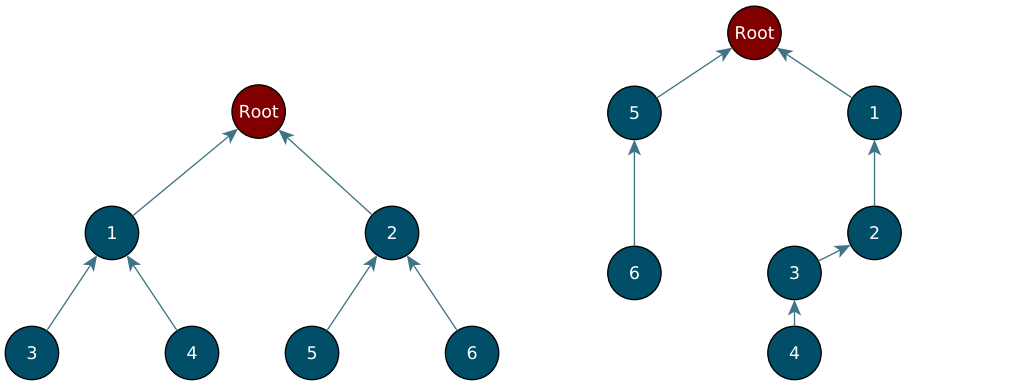


Figure 6.1: Generic Markov chain.

chain and hence, the end to end delay for this chosen value can be computed. In the next section we present the results done for a X-MAC and RPL scenario where we generalize a Markov chain model for estimating the end to end delay in a multi-hops scenario.

6.4 X-MAC/RPL Scenarios

In this section we show how the Markov chain generalisation is carried out for an scenario using X-MAC and RPL protocols. We show results for two differents network topologies.



(a) Tree topology: 2 hops

(b) Tree topology: 4 hops

Figure 6.2: Two different topologies.

6.4.1 Scenarios

In order to show the suitability of our approach we have set two differents scenario taking two differents topologies. Figures 6.2a and 6.2b show the two scenarios we have set up in order to apply the mentioned methodology. Contrarily to the scenario shown in 5.5.2, only the (OF0) objective function is taken into account in the RPL protocol.

6.4.2 Non-Linear Regression Over Transitions and States

The procedure detailed in this subsection is common for the two scenarios we have set for finding a general expression of the Markov chain. In order to generate both $\mathcal{P}_t$ and $\mathcal{D}_s$ necessary to apply the non-linear regression approach, we start by defining the following set of arrival rates:

$$\begin{aligned}\mathcal{A}_{rate} = \{&\lambda_4, \lambda_2, \lambda_1, \lambda_{0.5}, \lambda_{0.33}, \lambda_{0.25}, \lambda_{0.2}, \lambda_{0.16},\\ &\lambda_{0.14}, \lambda_{0.12}, \lambda_{0.11}, \lambda_{0.1}\}\end{aligned} \tag{6.15}$$

where λ_i means i packets per second. Then, by applying the methodology proposed in section 5.2, we generate the corresponding Markov chains $\mathcal{MC}_{\lambda_i}$ for each λ_i. In this way, for each node, we obtain twelve Markov chains from which we are able to generate both $\mathcal{P}_t$ and $\mathcal{D}_s$. This is done as described above, that is to say, for each transition $t^{(j)}$ presented in the set of Markov chains, we generate the corresponding set of probabilities $P^{t^{(j)}}$ by looking at probability value, for this transition, on each $\mathcal{MC}_{\lambda_i}$. The same procedure is applied in order to generate the set $D^{s^{(j)}}$ for each of the states except that, in this case, we consider the sojourn time on each state instead of probabilities. Then, non-linear regression is applied to each element of $\mathcal{P}_t$ in order to find the best fitting function modelling the transition's behavior. The same approach is applied over $\mathcal{D}_s$

elements in order to estimate the mathematical function modelling the behavior of sojourn delay within each state of the Markov chain. The result of this approach is a Markov chain whose transitions between states as well as the sojourn time on each states are functions in terms of λ. Having a function for each transition and state allow us to estimate, for any unknown λ, the corresponding value in terms of probability and sojourn delay respectively by simply evaluating each function in the chosen λ. Then, for each node in the network we obtain the corresponding Markov chain for a given λ. Figure 6.3 shows the corresponding Markov chain for an unknown λ value ($\lambda = 2.4$) for one of the network's nodes (Node 4 in Figure 6.2b). Transition's probability as well as the state's sojourn time were computed by evaluating each function for $\lambda = 2.4$. Figures 6.4a and 6.4b show the fitting functions found for a particular state and transition. Red points in 6.4a represent the transition's probability for each λ_i while blue line the parabolic approximation function f obtained after applying non-linear regression. In the same way, red points in 6.4b represent the sojourn time for the corresponding λ while blue line represents the logarithmic approximation function f obtained by means of non-linear regression. We only include two figures but the same approach is done for all transitions and states presented in the Markov chain.

6.4.3 End to End delay Estimation

From the non-linear regression approach previously described, we were able to obtain a general Markov chain where transition's probabilities and state's sojourn time are not values but functions of λ. Then, by means of the approach discribed in section 5.2.3, we are able to compute both one-hop and end to end delay in a wireless sensor network scenario.

6.4.4 Results

In order to evaluate the non-regression approach we make a comparision in terms of the end to end delay for different arrival rates λ_i. First of all, we computed the general Markov chain $\mathcal{GMC}_\lambda$ and then, for each λ_i we replaced this value in $\mathcal{GMC}_{\lambda=\lambda_i}$ obtaining then the corresponding Markov chain for λ_i. End to end delay distribution $\mathcal{D}_{\mathcal{GMC}_{\lambda=\lambda_i}}$ can then be computed for each λ_i. Then, for each λ_i, we found the empirical Markov chain $\mathcal{MC}_{\lambda_i}$ and the corresponding end to end delay distribution $\mathcal{D}_{\mathcal{MC}_{\lambda_i}}$. Comparision is done between both $\mathcal{D}_{\mathcal{GMC}_{\lambda=\lambda_i}}$ and $\mathcal{D}_{\mathcal{MC}_{\lambda_i}}$. In this way, we have set two scenarios in order to test the suitability of our approach. We start first by setting up a simple tree scenario where we compute the end to end delay in a 2-hops transmission scenario (Figure 6.2a). We then extend the analysis by considering a more complex tree scenario and we present results in terms of end to end delay in a 4-hops scenario (Figure 6.2b).

Tree Scenario: 2 hops

In this scenario, we consider eight arrival rate points, four belonging to the sample $\mathcal{A}_{rate}$: $\lambda_i = \{4, 2, 1, 0.2\}p/s$ and four unknown points $\lambda_i = \{1.33, 0.66, 0.28, 0.18\}p/s$. Dashed curves represent the empirical end to end delay $\mathcal{D}_{\mathcal{MC}_{\lambda_i}}$ while solid curves represent the end to end delay $\mathcal{D}_{\mathcal{GMC}_{\lambda=\lambda_i}}$ from node 4 to the root (Figure 6.2a). Figure 6.5 shows this comparision for those points belonging to the sample $\mathcal{A}_{rate}$ while Figure 6.6 shows the comparision for those unknown λ_i points.

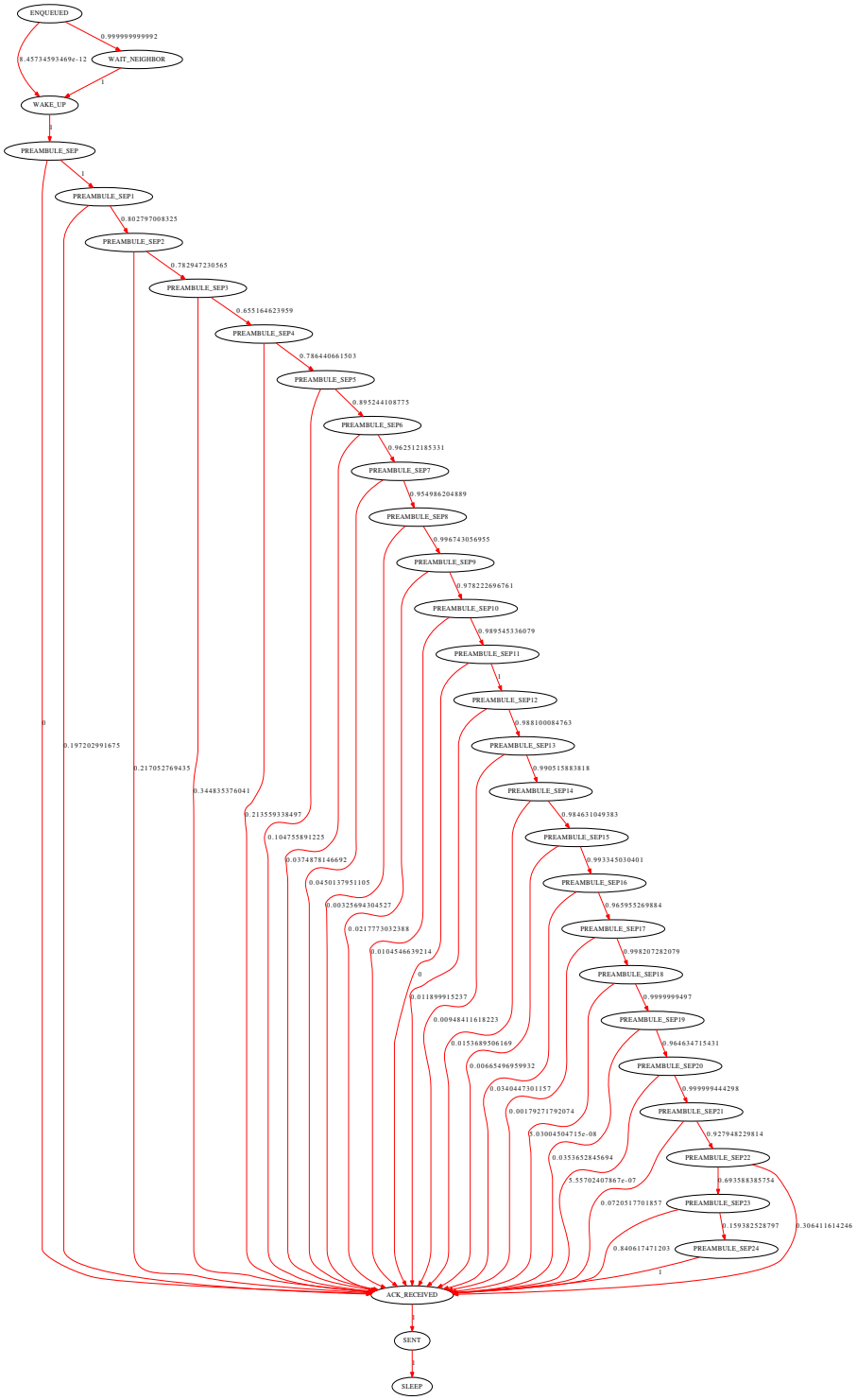
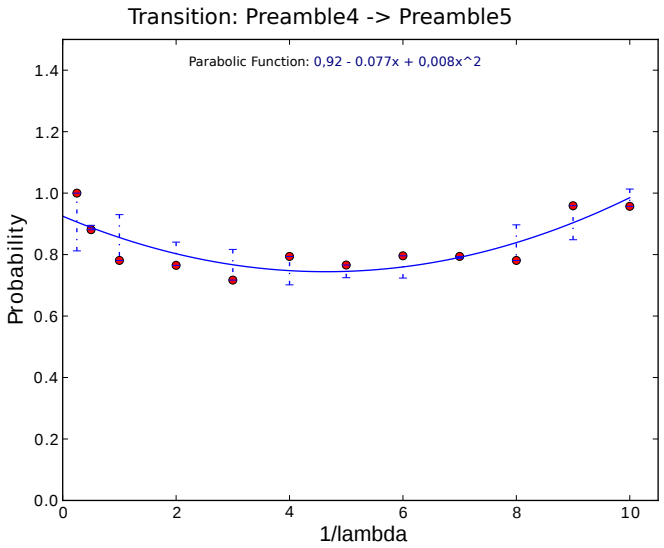


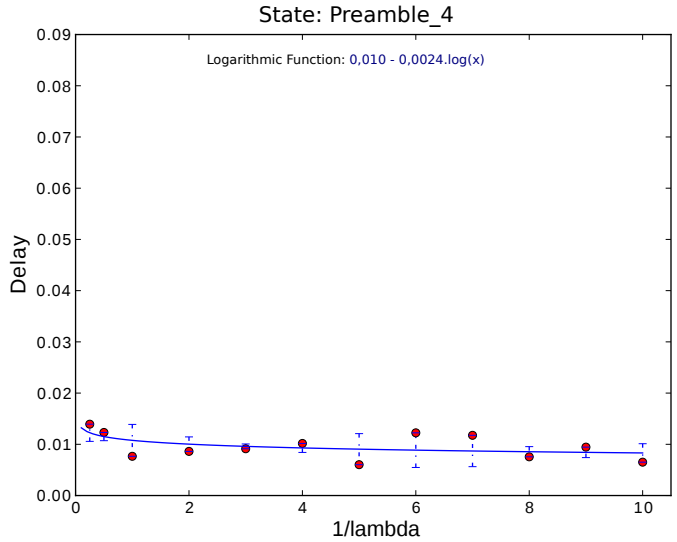
Figure 6.3: Markov chain ($\lambda = 2.4p/s$) for node 4 in Figure 6.2b.

Tree Scenario: 4 hops

In this scenario, we consider eight arrival rate points, four belonging to the sample $\mathcal{A}_{rate}$: $\lambda_i = \{4, 2, 1, 0.2\}p/s$ and four unknown points $\lambda_i = \{3, 1.33, 0.8, 0.18\}p/s$. Dashed curves represent



(a) Parabolic function modelling transition PREAMBLE_4 -> PREAMBLE_5.



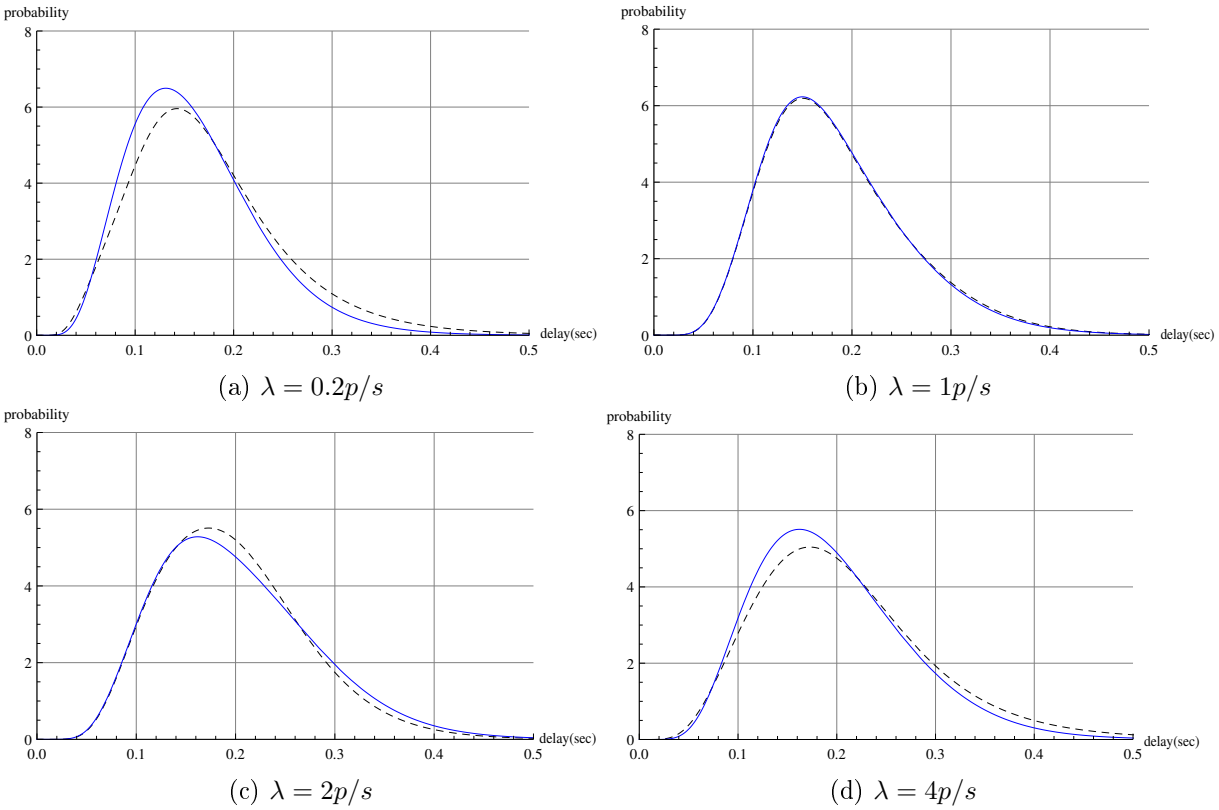
(b) Logarithmic function modelling sojourn time delay of state PREAMBLE_4.

Figure 6.4: Non-linear regression over transitions and states.

the empirical end to end delay $\mathcal{D}_{\mathcal{MC}_{\lambda_i}}$ while solid curves represent the end to end delay $\mathcal{D}_{\mathcal{GMC}_{\lambda=\lambda_i}}$ from node 4 to the root. Figure 6.7 shows this comparision for those points belonging to the sample $\mathcal{A}_{rate}$ while Figure 6.8 shows the comparision for those unknown λ_i points.

6.5 Discussions

Figures 6.5 and 6.6 show the comparision in terms of end to end delay distribution for a 2-hops transmission scenario while Figures 6.7 and 6.8 show the same comparision but for a 4-hops tree scenario. Roughly, we can say that our approach seems to be accurate for estimating the end



(a) $\lambda = 0.2p/s$ (b) $\lambda = 1p/s$

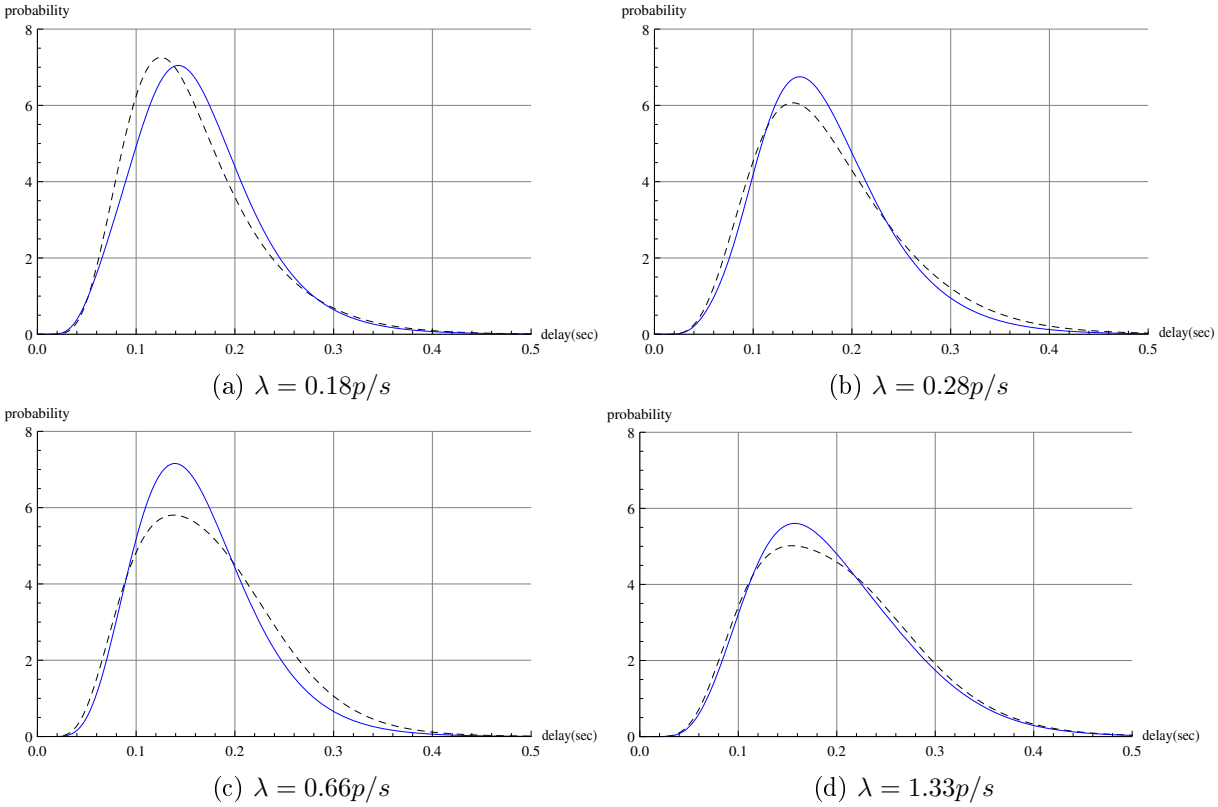
(c) $\lambda = 2p/s$ (d) $\lambda = 4p/s$

Figure 6.5: Probability density function comparision between non-linear regression approach and empirical results for known λ points.

to end delay distribution in a WSN. However, we can see somme differences for some of the curves when comparing both theoretical and empirical results. There are at least two reasons why this subtle difference is present. The first is regarding the empirical curve. This curve is the result of a single testbed execution for each of the lambda values and not an average of several ones. In order to improve the accuracy of the empirical curve what we can do is to generate a set of curves for a given lambda and then to take some kind of average curve to best represent the empirical execution of the scenario. Unfortunately, due to the lack of time, we could not carry out this experimentation. Secondly, we have explained in section 6.4.2 that the non-linear regression approach approximates the function that best fits a set of empirical points (for both transition's probability and state's sojourn delay). As we can see from Figures 6.4a and 6.4b, curves sometimes overestimates and sometimes underestimates the points. That means for instances that in some cases the function modeling the sojourn delay in a particular state will under/over estimate it, leading to not enough accurate estimation of the local delay and hence the whole end to end delay. Optimisation in this direction can be done in order to improve this estimation. As an example, we can consider a new set of potential functions or even modify the existing ones in order to best fit the empirical points.

6.6 Conclusion

In chapter 5, we have presented a contribution combining measurements and analytic approaches for finding a Markov chain modelling a WSN considering different MAC protocols. However,



(a) $\lambda = 0.18p/s$ (b) $\lambda = 0.28p/s$

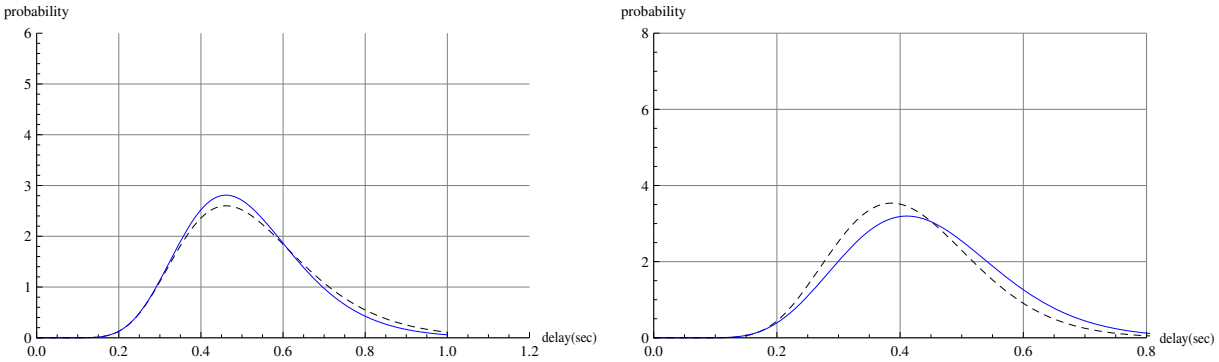
(c) $\lambda = 0.66p/s$ (d) $\lambda = 1.33p/s$

Figure 6.6: Probability density function comparision between non-linear regression approach and empirical results for unknown λ points.

the model we have obtained is traffic-dependant so changing the packet arrival rate will give a different Markov model as result. In this chapter we have presented an approach based on the non-linear regression technique for generalising this Markov chain where both transition's probabilities and state's sojourn time are not values but functions of the packet arrival rate λ. The main contribution of this chapter is the generalisation of a Markov chain model for whatever packet arrival rates from which we can estimate the end to end delay distribution for a multi-hop transmission scenario, even for unknown packet arrival rates. Contrarily to existing Markov chain models, where there are assumptions in terms of the distribution of the packet arrival (basically Markovian traffic), our approach does not make any assumption regarding this distribution. Since we obtain a local Markov chain for each node belonging to the network, end to end delay is then computed based on the local delay distribution found from local Markov chain models. This presents some advantages for evaluating the end to end delay for different scenarios:

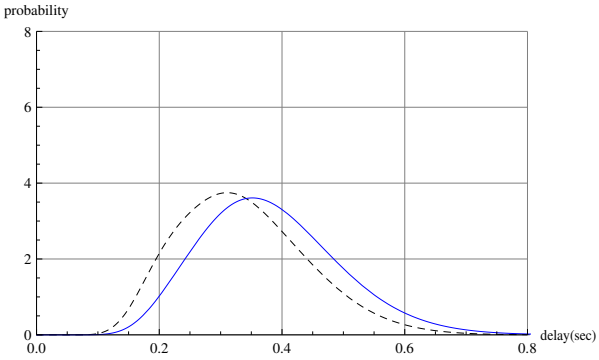
- We can set different arrival rates to each of the nodes belonging to the network in order to find the end to end delay in a multi-hop scenario considering different arrival rates.

- Since end to end delay is computed based on local Markov chain models, our approach can be useful for evaluating the delay for different routing scenarios, for instance, by setting differents path probabilities from nodes to the root and evaluating each of the scenarios in terms of the delay.

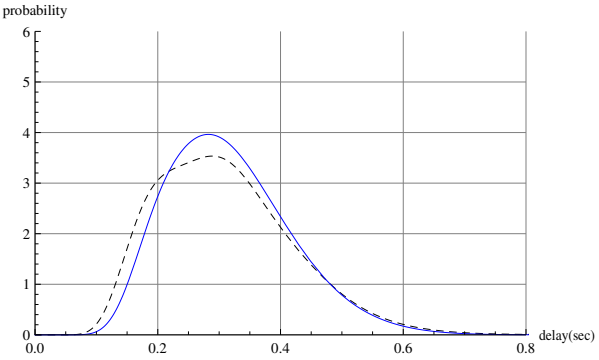


(a) $\lambda = 4p/s$

(b) $\lambda = 2p/s$



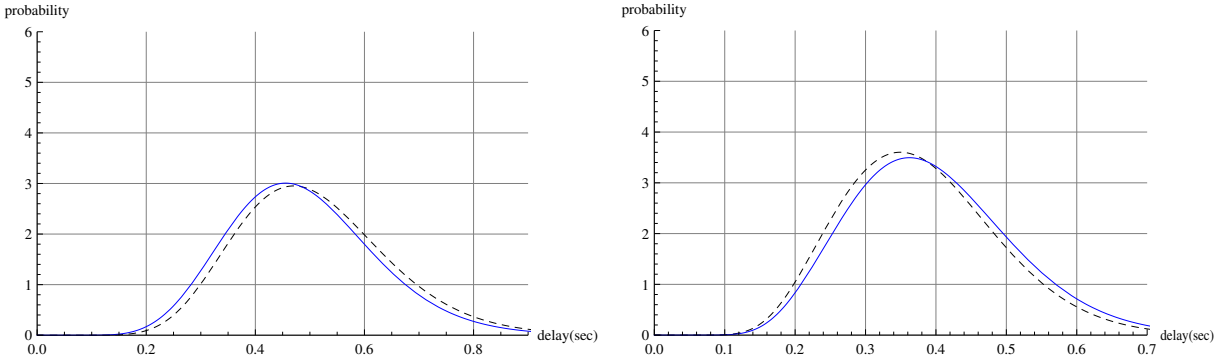
(c) $\lambda = 1p/s$



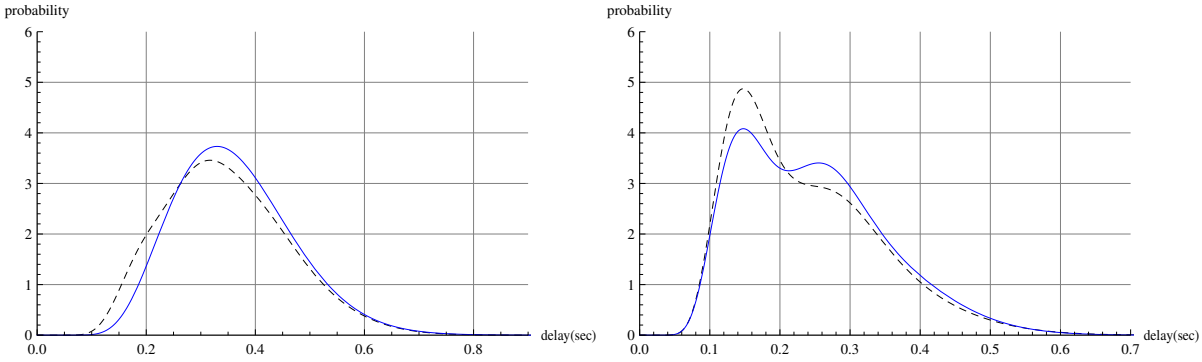
(d) $\lambda = 0.5p/s$

Figure 6.7: Probability density function comparision between non-linear regression approach and empirical results for known λ points.

Most of the existing Markov models found in the literature were conceived for the IEEE 802.15.4 MAC protocol. By means of our approach, we are able to find a Markov chain model for any of the underlying MAC protocol as we have done for the IEEE 802.15.4, ContikiMAC and X-MAC in sections 5.3, 5.4 and 5.5 respectively.



(a) $\lambda = 3p/s$ (b) $\lambda = 1.33p/s$



(c) $\lambda = 0.8p/s$ (d) $\lambda = 0.18p/s$

Figure 6.8: Probability density function comparision between non-linear regression approach and empirical results for unknown λ points.

Chapter 7

Conclusions & Future Work

Contents

7.1 Conclusions

In wireless sensor networks, there are many real-time applications that require bounded delay. For these applications, having an estimation of the end to end delay is crucial in order to achieve the application goals and to guarantee a proper operation of the network. However, because of the randomness in wireless communication and the low duty-cycle of nodes in WSNs, the estimation of the end to end delay is challenging. In this thesis, our objective is to be able to estimate the end to end delay in a multi-hop tranmission scenario in a WSN. We are interested in finding a probabilistic estimation where the end to end delay should not exceed the given threshold with certain probability.

We have presented an overview of the existing approaches found in the literature for estimating the end to end delay in WSNs. We have seen that most of the simulation models are simplification of the real world and therefore, this approach is not accurate enough for estimating performance parameters. We have also seen that, due to the difficulty to achieve global synchronisation in large-scale sensor networks, the measurement-based approach for estimating the end to end delay is extremely challenging. Analytical solutions found in the literature for modelling the behavior of the MAC layer protocols for estimating the local delay within a node, were also presented. Due to the complexity in mathematical terms for modelling the behavior of MAC protocols in real conditions, most of these analytical approaches simplify the reality by considering ideal channel conditions, by simplifying the queue model on each node, or even not considering the issues introduced by the underlying operating system. Besides, most of the proposed analytical solutions are conceived for the standard IEEE 802.15.14 MAC protocol and limited to a star topology. Hence, the whole delay is estimated in terms of the one-hop delay from a source node to the coordinator located at the center of the topology. The extension of these analytical models for considering a multi-hop topology is not possible since the packet distribution arriving to intermediate nodes is not known. One of the most achieved Markov model conceived for WSNs was presented by Misic et al. [Misic and Misic, 2008] for the IEEE 802.15.4 MAC protocol. Although this model captures the most important factors in WSN, the extension of this model for considering multi-hop scenarios is not possible. Besides, with our testbed experiments measured using TelosB, TinyOS

and an implementation of the IEEE 802.15.4 MAC protocol, we have seen that the underlying operating system introduces non-negligible delays that are not taken into account for these analytical models, leading to huge gaps between analytical results and measurement in terms of delay.

In this thesis, we have presented a novel approach for modelling the behavior of a WSN overcoming the main problems found in the literature. From the execution traces of a given MAC protocol, we are able to find a Markov chain for each node modelling the behavior of the underlying MAC protocol running on a platform. This Markov model allows us to estimate the one hop delay for each node, that is to say, the delay from the moment a packet arrives to a node until the moment the corresponding acknowledgement is received. Then, by means of a frequency domain analysis, we are able to estimate the end to end delay in a multi-hop transmission scenario. Since our approach is based on the analysis of MAC protocol execution traces, the first Markov chain we have obtained were traffic-dependant, that is to say that, changing the packet arrival rate may lead to a different Markov model. In other words, for a set of packet arrival rate λ it is necessary to run experiments for each of the λ arrival rates in order to obtain the corresponding Markov chain models. In order to overcome this problem, we have proposed an approach based on non-linear regression techniques for generalising this Markov chain where both transition's probabilities and state's sojourn time are no longer values but functions of the packet arrival rate λ. This generalisation allows us to discover a Markov chain model independent of the packet arrival rate from which we can estimate the end to end delay distribution for a multi-hop transmission scenario. Through our experience, one interesting finding is that in a WSN where MAC and routing protocols interplay, we only need to instrument the MAC protocol execution to achieve the end to end delay estimation. Routing protocol dynamics only contribute to modify the transition probabilities between nodes. This makes our approach simple and applicable to a larger class of other types of networks.

In summary, the contributions of this thesis can be enumerated as follows:

- We propose a new methodology to discover a Markov chain modelling the behavior of nodes in a WSN for a given MAC protocol. Since this Markov chain is found from the MAC protocol traces analysis, factors like the impact of the underlying operating system are taken into account, an issue not considered in existing analytical models.
- Contrarily to the existing approaches where the analysis is done for a particular MAC protocol (the IEEE 802.15.4 MAC protocol), our methodology is applicable for any underlying MAC protocols, provided that the source code is available in order to be able to instrument it, which is the case for most of existing MAC protocols.
- Contrarily to the existing Markov chain models, where the assumptions in terms of the distribution of the packet arrival for modelling the node behavior must be considered (Markovian traffic), our approach does not make any assumption regarding this distribution. Hence, this approach allows to model the behavior of intermediate nodes without strong assumptions on the packet arrival distribution.
- Based on this Markov chain, we are able to estimate the one hop delay for each of the nodes in the network (even for intermediate nodes) from the moment a packet arrives to the node until the corresponding acknowledgement is received.
- By means of a frequency domain analysis proposed in [He et al., 2010a], we are able to extend the analysis for estimating the end to end delay in a multi-hop transmission scenario,

the core of this thesis.

- We also propose an approach based on non-linear regression techniques to find a general Markov chain model that allows us to estimate the end to end delay for unknown arrival rates without executing the experimentation. This presents an advantage with regard to the measurement approach where, for unknown λ's, it is necessary to launch multiple experimentations in order to obtain the desired result.

7.2 Future Work

The approach proposed in this thesis for modelling a WSN can be extended in several directions. Experimentations done along this thesis for validating the approach consider small scale topologies in a real testbed. We plan to extend the experimentations done so far to larger scale networks in order to explore how scalable our approach is.
As we have seen previously, the Markov chain generalisation approach proposed is done in terms of the packet arrival rate. We have seen that for this general Markov chain, transitions are not probabilities but functions of the arrival rate λ. Another extension of this approach may be to also consider some other parameters (such as the number of neighbors for a given node) in such a way that the transitions within the obtained Markov model would be multi-variable instead of single-variable functions. This extension can be achieved by considering a multivariable non-linear regression.

Chapter 8

List of Publications

Extracting Markov Chain Models from Protocol Execution Traces for End to End Delay Evaluation in Wireless Sensor Networks.
11th IEEE World Conference in Factory Communication Systems - May 27-29, 2015. Palma de Mallorca, Spain.

Towards Performance Analysis of Wireless Sensor Networks Using Process Mining Techniques.
19th IEEE ISCC - Funchal, Madeira, Portugal, June 23-26, 2014.

Modelling and Performance Analysis of Wireless Sensor Networks Using Process Mining Techniques: ContikiMAC use case.
IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2014). Marina Del Rey, CA, USA, May 26-28, 2014

Measurement-based Analysis of the Effect of Duty Cycle in IEEE 802.15.4 MAC Performance.
1st International Workshop on Compressive Sensing in Cyber-Physical Systems (CSCPS) in conjunction with MASS2013. Hangzhou, China, October 14-16, 2013.

On the Gap Between Mathematical Modeling and Measurement Analysis for Performance Evaluation of the 802.15.4 MAC Protocol.
The 12th International Workshop on Real-Time Networks in conjunction with euromicro RTS. July 9th, 2013, Paris, France.

Combining Analytical and Simulation Approaches for Estimating End-to-End Delay in Multi-hop Wireless Networks.
WiSARN 2012 in conjuction with DCOSS 2012. May 16 - 18, 2012, Hangzhou, Chine

Bibliography

[EWS, 2007] (2007). Earthquake Early Warning System.

[802, 2007] (2007). Ieee 802.15.4. Technical report.

[Abdelzaher et al., 2004] Abdelzaher, T., Prabh, S., and Kiran, R. (2004). On real-time capacity limits of multihop wireless sensor networks. In *Real-Time Systems Symposium, 2004. Proceedings. 25th IEEE International*, pages 359–370.

[Allen, 1990] Allen, A. O. (1990). *Probability, statistics and queueing theory - with computer science applications (2. ed.)*, chapter 5. Academic Press.

[Beaudaux et al., 2014] Beaudaux, J., Gallais, A., Montavont, J., Noel, T., Roth, D., and Valentin, E. (2014). Thorough empirical analysis of x-mac over a large scale internet of things testbed. *Sensors Journal, IEEE*, 14(2):383–392.

[Bianchi, 2006] Bianchi, G. (2006). Performance analysis of the ieee 802.11 distributed coordination function. *IEEE J.Sel. A. Commun.*, 18(3):535–547.

[Bisnik and Abouzeid, 2006] Bisnik, N. and Abouzeid, A. (2006). Queuing network models for delay analysis of multihop wireless ad hoc networks. In *Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*, IWCMC '06, pages 773–778, New York, NY, USA. ACM.

[Buettner et al., 2006] Buettner, M., Yee, G. V., Anderson, E., and Han, R. (2006). X-mac: A short preamble mac protocol for duty-cycled wireless sensor networks. SenSys '06, pages 307–320, New York, NY, USA. ACM.

[Burchard et al., 2006] Burchard, A., Liebeherr, J., and Patek, S. (2006). A min-plus calculus for end-to-end statistical service guarantees. *Information Theory, IEEE Transactions on*, 52(9):4105–4114.

[Chatzimisios et al., 2002] Chatzimisios, P., Vitsas, V., and Boucouvalas, A. (2002). Throughput and delay analysis of ieee 802.11 protocol. In *Networked Appliances, 2002. Liverpool. Proceedings. 2002 IEEE 5th International Workshop on*, pages 168–174.

[Chaudhary and Waghmare, 2012] Chaudhary, D. and Waghmare, L. (2012). Quality of service analysis in wireless sensor network by controlling end-to-end delay. In *Industrial Electronics and Applications (ICIEA), 2012 7th IEEE Conference on*, pages 703–708.

[Cruz, 1991] Cruz, R. (1991). A calculus for network delay. i. network elements in isolation. *Information Theory, IEEE Transactions on*, 37(1):114–131.

[Demirkol et al., 2006] Demirkol, I., Ersoy, C., and Alagoz, F. (2006). Mac protocols for wireless sensor networks: a survey. *Communications Magazine, IEEE*, 44(4):115–121.

[Despaux et al., 2014a] Despaux, F., Song, Y., and Lahmadi, A. (2014a). Modelling and performance analysis of wireless sensor networks using process mining techniques: Contikimac use case. In *IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS 2014, Marina Del Rey, CA, USA, May 26-28, 2014*, pages 225–232.

[Despaux et al., 2013] Despaux, F., Song, Y.-Q., and Lahmadi, A. (2013). On the gap between mathematical modeling and measurement analysis for performance evaluation of the 802.15.4 mac protocol. In *12th International Workshop on Real Time Networks RTN'13*.

[Despaux et al., 2014b] Despaux, F., Song, Y.-Q., and Lahmadi, A. (2014b). Towards performance analysis of wireless sensor networks using process mining techniques. In *ISCC 2014*.

[Despaux et al., 2015] Despaux, F., Song, Y.-Q., and Lahmadi, A. (2015). Extracting markov chain models from protocol execution traces for end to end delay evaluation in wireless sensor networks [accepted]. In *11th IEEE World Conference on Factory Communication Systems*.

[Deuflhard, 2011] Deuflhard, P. (2011). Least squares problems: Gauss-newton methods. In *Newton Methods for Nonlinear Problems*, volume 35 of *Springer Series in Computational Mathematics*, pages 173–231. Springer Berlin Heidelberg.

[Du et al., 2010] Du, W., Navarro, D., Mieyeville, F., and Gaffiot, F. (2010). Towards a taxonomy of simulation tools for wireless sensor networks. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, SIMUTools '10, pages 52:1–52:7, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Dunkels, 2011] Dunkels, A. (2011). The ContikiMAC Radio Duty Cycling Protocol. Technical Report T2011:13, Swedish Institute of Computer Science.

[Durisic et al., 2012] Durisic, M., Tafa, Z., Dimic, G., and Milutinovic, V. (2012). A survey of military applications of wireless sensor networks. In *Embedded Computing (MECO), 2012 Mediterranean Conference on*, pages 196–199.

[El-Hoiydi and Decotignie, 2004] El-Hoiydi, A. and Decotignie, J.-D. (2004). Wisemac: an ultra low power mac protocol for the downlink of infrastructure wireless sensor networks. In *Computers and Communications, 2004. Proceedings. ISCC 2004. Ninth International Symposium on*, volume 1, pages 244–251 Vol.1.

[Eriksson et al., 2009] Eriksson, J., Österlind, F., Finne, N., Tsiftes, N., Dunkels, A., Voigt, T., Sauter, R., and Marrón, P. J. (2009). Cooja/mspsim: Interoperability testing for wireless sensor networks. In *Proceedings of the 2Nd International Conference on Simulation Tools and Techniques*, Simutools '09, pages 27:1–27:7, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Fambon et al., 2014] Fambon, O., Fleury, E., Harter, G., Pissard-Gibollet, R., and Saint-Marcel, F. (2014). FIT IoT-LAB tutorial: hands-on practice with a very large scale testbed tool for the Internet of Things.

[Fidler, 2006] Fidler, M. (2006). An end-to-end probabilistic network calculus with moment generating functions. In *in Proc. IEEE 14th International Workshop on Quality of Servic (IWQoS*, pages 261–270.

[Gupta and Shroff, 2009] Gupta and Shroff (2009). Delay analysis for multi-hop wireless networks. In *INFOCOM 2009, IEEE*, pages 2356–2364.

[Hadzi-Velkov and Spasenovski, 2003] Hadzi-Velkov, Z. and Spasenovski, B. (2003). Saturation throughput - delay analysis of ieee 802.11 dcf in fading channel. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 1, pages 121–126 vol.1.

[Hauer, 2009] Hauer, J.-H. (2009). Tkn15.4: An ieee 802.15.4 mac implementation for tinyos 2. TKN Technical Report Series TKN-08-003, Telecommunication Networks Group, Technical University Berlin.

[He et al., 2010a] He, W., Liu, X., Zheng, L., and Yang, H. (2010a). Reliability calculus: A theoretical framework to analyze communication reliability. In *Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems*, ICDCS '10, pages 159–168, Washington, DC, USA. IEEE Computer Society.

[He et al., 2010b] He, W., Liu, X., Zheng, L., and Yang, H. (2010b). Reliability calculus: A theoretical framework to analyze communication reliability. ICDCS '10, pages 159–168, Washington, DC, USA. IEEE Computer Society.

[Huang et al., 2009] Huang, Y.-K., Pang, A.-C., and Hung, H.-N. (2009). A comprehensive analysis of low-power operation for beacon-enabled IEEE 802.15.4 wireless networks. *IEEE Transactions on Wireless Communications*, 8(11):5601–5611.

[Issariyakul and Hossain, 2008] Issariyakul, T. and Hossain, E. (2008). *Introduction to Network Simulator NS2*. Springer Publishing Company, Incorporated, 1 edition.

[Jurcik et al., 2007] Jurcik, P., Koubaa, A., Alves, M., Tovar, E., and Hanzalek, Z. (2007). A simulation model for the IEEE 802.15.4 protocol: Delay/throughput evaluation of the GTS mechanism. In *15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2007), October 24-26, 2007, Istanbul, Turkey*, pages 109–116.

[Kredo and Mohapatra, 2007] Kredo, II, K. and Mohapatra, P. (2007). Medium access control in wireless sensor networks. *Comput. Netw.*, 51(4):961–994.

[Kuntz, 2010] Kuntz, R. (2010). *Medium Access Control Facing the Dynamics of Wireless Sensor Networks*. Theses.

[Langendoen, 2007] Langendoen, K. (2007). Medium access control in wireless sensor networks.

[Lehoczky, 1997] Lehoczky, J. (1997). Real-time queueing network theory. In *Real-Time Systems Symposium, 1997. Proceedings., The 18th IEEE*, pages 58–67.

[Levis et al., 2003] Levis, P., Lee, N., Welsh, M., and Culler, D. (2003). Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 126–137, New York, NY, USA. ACM.

[Liu et al., 2013] Liu, K., Ma, Q., Liu, H., Cao, Z., and Liu, Y. (2013). End-to-end delay measurement in wireless sensor networks without synchronization. In *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*, pages 583–591.

[Lucio et al., 2003] Lucio, G. F., Paredes-farrera, M., Jammeh, E., Fleury, M., and Reed, M. J. (2003). Opnet modeler and ns-2: Comparing the accuracy of network simulators for packet-level analysis using a network testbed. In *In 3rd WEAS International Conference on Simulation, Modelling and Optimization (ICOSMO*, pages 700–707.

[M. Xie, 2009] M. Xie, M. H. (2009). Towards an end-to-end delay analysis of wireless multi-hop networks. *Ad Hoc Networks*, 7:849–861.

[Misic and Misic, 2008] Misic, J. and Misic, V. (2008). *Wireless Personal Area Networks: Performance, Interconnection, and Security with IEEE 802.15.4*. Wiley Publishing.

[Neugebauer et al., 2005] Neugebauer, M., Plonnigs, J., and Kabitzsch, K. (2005). A new beacon order adaptation algorithm for IEEE 802.15. 4 networks. In *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, page 302–311.

[Norris, 1999] Norris, J. (1999). *Markov Chains*. Cambridge series in statistical and probabilistic mathematics. Cambridge University Press.

[Park et al., 2009] Park, P., Di Marco, P., Soldati, P., Fischione, C., and Johansson, K. (2009). A generalized markov chain model for effective analysis of slotted ieee 802.15.4. In *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, pages 130–139.

[Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):pp. 1065–1076.

[Polastre et al., 2004] Polastre, J., Hill, J., and Culler, D. (2004). Versatile low power media access for wireless sensor networks. In *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 95–107, New York, NY, USA. ACM.

[Raptis et al., 2006] Raptis, P., Banchs, A., and Paparrizos, K. (2006). A-simple-and-effective-delay-distribution-analysis-for-ieee-802.11. In *Personal, Indoor and Mobile Radio Communications, 2006 IEEE 17th International Symposium on*, pages 1–5.

[rong Chen et al., 2007] rong Chen, B., Peterson, G., Mainland, G., and Welsh, M. (2007). Livenet: Using passive monitoring to reconstruct sensor network dynamics. Technical report.

[Serna Oliver and Fohler, 2009] Serna Oliver, R. and Fohler, G. (2009). A proposal for a notion of timeliness in wireless sensor networks. In *Proceedings of the 8th International Workshop on Real-Time Networks RTN'09*, Dublin, Ireland.

[Shu et al., 2006] Shu, F., Sakurai, T., Vu, H. L., and Zukerman, M. (2006). Optimizing the IEEE 802.15. 4 MAC. In *TENCON 2006. 2006 IEEE Region 10 Conference*, page 1–4.

[Simon et al., 2003] Simon, G., Volgyesi, P., Maroti, M., and Ledeczi, A. (2003). Simulation-based optimization of communication protocols for large-scale wireless sensor networks. In *Proceedings of IEEE Aerospace Conference*, volume 3, pages 1339–1346, Big Sky, MT, USA.

[Singh et al., 2008] Singh, C., Vyas, O., and Tiwari, M. (2008). A survey of simulation in sensor networks. In *Computational Intelligence for Modelling Control Automation, 2008 International Conference on*, pages 867–872.

[Suriyachai et al., 2009] Suriyachai, P., Roedig, U., and Scott, A. (2009). Implementation of a MAC protocol for QoS support in wireless sensor networks. In *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*, page 1–6.

[van Dam and Langendoen, 2003] van Dam, T. and Langendoen, K. (2003). An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 171–180, New York, NY, USA. ACM.

[Varga and Hornig, 2008] Varga, A. and Hornig, R. (2008). An overview of the omnet++ simulation environment. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Wang et al., 2012] Wang, Y., Vuran, M. C., and Goddard, S. (2012). Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. *IEEE/ACM*, 20(1):305–318.

[Winter et al., 2012] Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, J., and Alexander, R. (2012). RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard).

[Xie et al., 2014] Xie, H., Zhang, G., Su, D., Wang, P., and Zeng, F. (2014). Performance evaluation of rpl routing protocol in 6lowpan. In *Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on*, pages 625–628.

[Yahya and Ben-Othman, 2009] Yahya, B. and Ben-Othman, J. (2009). Towards a classification of energy aware mac protocols for wireless sensor networks. *Wireless Communications and Mobile Computing*, 9(12):1572–1607.

[Yang and Heinzelman, 2012] Yang, O. and Heinzelman, W. B. (2012). Modeling and performance analysis for duty-cycled mac protocols with applications to s-mac and x-mac. *IEEE Transactions on Mobile Computing*, 11(6):905–921.

[Ye et al., 2009] Ye, D., Gong, D., and Wang, W. (2009). Application of wireless sensor networks in environmental monitoring. In *Power Electronics and Intelligent Transportation System (PEITS), 2009 2nd International Conference on*, volume 1, pages 205–208.

[Ye et al., 2002] Ye, W., Heidemann, J., and Estrin, D. (2002). An energy-efficient MAC protocol for wireless sensor networks. In *INFOCOM 2002.*, volume 3, pages 1567–1576.

[Zhou and Mitchell, 2010] Zhou, J. and Mitchell, K. (2010). A scalable delay based analytical framework for csma/ca wireless mesh networks. *Comput. Netw.*, 54:304–318.

[Zuniga and Krishnamachari, 2004] Zuniga, M. and Krishnamachari, B. (2004). Analyzing the transitional region in low power wireless links. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, page 517 – 526.

Résumé

Dans cette thèse, nous proposons une nouvelle approche pour modéliser et estimer les délais de bout-en-bout dans les réseaux de capteurs sans-fil (WSN). Notre approche combine les approches analytiques et expérimentales pour inférer un modèle Markovien modélisant le comportement d'un protocole de contrôle d'accès au médium (MAC) exécuté sur les noeuds d'un réseau de capteurs. À partir de ce modèle Markovien, le délai de bout en bout est ensuite obtenu par une approche analytique basée sur une analyse dans le domaine fréquentiel pour calculer la probabilité de distribution de délais pour un taux d'arrivée spécifique. Afin d'obtenir une estimation du délai de bout en bout, indépendamment du trafic en entrée, la technique de régression non-linéaire est utilisée sur un ensemble d'échantillons limités. Cette approche nous a permis de contourner deux problèmes: 1) la difficulté d'obtenir un modèle Markovien du comportement d'un protocole MAC en tenant compte de son implémentation réelle, 2) l'estimation du délai de bout-en-bout d'un WSN multi-sauts. L'approche a été validée sur un testbed réel (IOT-LAB) et pour plusieurs protocoles (X-MAC, ContikiMAC, IEEE 802.15.4) ainsi que pour un protocole de routage (RPL).

Mots-clés: réseaux de capteurs, protocoles MAC, délai de bout en bout.

Abstract

In this thesis, we propose an approach that combines both measurements and analytical approaches for infering a Markov chain model from the MAC protocol execution traces in order to be able to estimate the end to end delay in multi-hop transmission scenarios. This approach allows capturing the main features of WSN. Hence, a suitable Markov chain for modelling the WSN is infered. By means of an approach based on frequency domain analysis, end to end delay distribution for multi-hop scenarios is found. This is an important contribution of our approach with regard to existing analytical approaches where the extension of these models for considering multi-hop scenarios is not possible due to the fact that the arrival distribution to intermediate nodes is not known. Since local delay distribution for each node is obtained by analysing the MAC protocol execution traces for a given traffic scenario, the obtained model (and therefore, the whole end to end delay distribution) is traffic-dependant. In order to overcome this problem, we have proposed an approach based on non-linear regression techniques for generalising our approach in terms of the traffic rate. Results were validated for different MAC protocols (X-MAC, ContikiMAC, IEEE 802.15.4) as well as a well-known routing protocol (RPL) over real test-beds (IOT-LAB).

Keywords: wireless sensor networks, MAC protocols, end to end delay.